TUGAS AKHIR – KI141502

IMPLEMENTASI MODUL MONITORING DAN KLASIFIKASI TANGGAPAN ADUAN DENGAN METODE NAÏVE BAYES PADA APLIKASI PERANGKAT BERGERAK SUARA WARGA KOTA KEDIRI

MAHARDHIKA RIZKI BAGASKORO
NRP 5111 100 188

Dosen Pembimbing
Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dwi Sunaryono, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – KI141502

IMPLEMENTATION OF MONITORING AND COMPLAINT'S RESPONSES CLASSIFICATION MODULE WITH NAÏVE BAYES METHOD ON SUARA WARGA KOTA KEDIRI MOBILE APPLICATION

MAHARDHIKA RIZKI BAGASKORO
NRP 5111 100 188

Advisor
Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dwi Sunaryono, S.Kom., M.Kom.

INFORMATICS DEPARTMENT
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

LEMBAR PENGESAHAN

IMPLEMENTASI MODUL MONITORING DAN KLASIFIKASI TANGGAPAN ADUAN DENGAN METODE NAÏVE BAYES PADA APLIKASI PERANGKAT BERGERAK SUARA WARGA KOTA KEDIRI

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Algoritma dan Pemrograman
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
MAHARDHIKA RIZKI BAGASKORO
NRP. 5111 100 188

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr.tech. Ir. R. V. Hari Ginardi, M.Sc. .....
   NIP: 19650518 199203 1 003 (Pembimbing 1)

2. Dwi Sunaryono, S.Kom., M.Kom. .....
   NIP: 19720528 199702 1 001 (Pembimbing 2)

SURABAYA
JUNI, 2015

IMPLEMENTASI MODUL MONITORING DAN KLASIFIKASI TANGGAPAN ADUAN DENGAN METODE NAÏVE BAYES PADA APLIKASI PERANGKAT BERGERAK SUARA WARGA KOTA KEDIRI

Nama Mahasiswa : Mahardhika Rizki Bagaskoro
NRP : 5111 100 188
Jurusan : Teknik Informatika FTIF-ITS
Dosen Pembimbing 1 : Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dosen Pembimbing 2 : Dwi Sunaryono, S.Kom., M.Kom.

Abstrak

Melakukan monitoring terhadap kinerja departemen pada suatu pemerintahan merupakan kegiatan yang penting dilakukan setiap saat oleh pemerintah untuk menjaga kualitas pelayanan kepada masyarakat tetap terjaga. Kinerja suatu departemen dapat dinilai dari tanggapan yang diberikan untuk aduan yang mereka terima. Hal seperti ini memungkinkan departemen selalu menanggapi setiap aduan dengan sembarangan agar mendapatkan penilaian yang baik, sehingga diperlukan sebuah metode untuk mengelompokkan tanggapan yang diberikan oleh departemen tertentu terhadap aduan yang diterimananya.

Pada Tugas Akhir ini akan dibangun suatu modul aplikasi pada perangkat bergerak yang dapat melakukan kegiatan monitoring serta mengelompokkan tanggapan-tanggapan departemen secara otomatis menggunakan metode Naïve Bayes. Metode ini mengategorikan objek baru berdasarkan atribut dan sampel data training. Naïve Bayes memanfaatkan nilai probabilitas semua tanggapan dan setiap kategori untuk mengelompokkan tanggapan. Modul aplikasi ini dibangun pada sistem operasi Android.

Modul aplikasi yang telah dikembangkan diuji berdasarkan fungsionalitasnya. Pengujian dilakukan melalui skenario yang

mencerminkan fitur aplikasi. Berdasarkan uji coba terhadap klasifikasi yang dilakukan menggunakan 36 tanggapan yang dihimpun dari tanggapan pada tanggal 5 Januari – 20 Februari 2015 yang diambil dari database Suara Warga Kota Kediri, menunjukkan bahwa metode Naïve Bayes mampu melakukan klasifikasi tanggapan berbahasa Indonesia dengan tingkat akurasi mencapai 80,56%. Hasil pengujian menunjukkan modul aplikasi dapat bekerja dengan baik.

Kata kunci: Android, klasifikasi, Naïve Bayes, monitoring, perangkat bergerak.

IMPLEMENTATION OF MONITORING AND COMPLAINT'S RESPONSES CLASSIFICATION MODULE WITH NAÏVE BAYES METHOD ON SUARA WARGA KOTA KEDIRI MOBILE APPLICATION

Student's Name	**: Mahardhika Rizki Bagaskoro**
Student's ID	**: 5111 100 188**
Department	**: Informatics FTIF-ITS**
Advisor 1	**: Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.**
Advisor 2	**: Dwi Sunaryono, S.Kom., M.Kom.**

Abstract

Monitoring department's performance on a government agencies is an important activity to do every time by the government in order to maintain their service quality towards its people. Department's performance can be rated from their responses towards complaints they received. These things may enable a department to always response a complaint carelessly in order to get a good score, therefore it's needed to make a method that categorize complaint's response given by a department.

This final project aims to build a module that works on a mobile devices in which can do monitoring activities and classifying department's responses automatically using Naïve Bayes method. This method will categorize new objects based on the attributes and training data samples. Naïve bayes utilize the probability value of all responses and each category to group those responses. This module built on Android.

The developed module tested based on its functionality. The testing done trough several scenarios which reflects the feature of the application module. Based on a test on classification performed using 36 responses gathered from January 5th until February 20th 2015 taken from Suara Warga Kota Kediri database, suggesting that Naïve Bayes capable of classifying Indonesian language

responses with an accuracy of 80,56%. The test results show that application run properly.

Keywords: Android, classification, Naïve Bayes, monitoring, mobile device.

KATA PENGANTAR

Puji syukur kehadirat Allah SWT atas segala kekuatan, bimbingan, berkah, rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir ini. Pada kesempatan ini, Penulis ingin menyampaikan penghormatan dan rasa terima kasih sedalam-dalamnya kepada pihak-pihak yang telah membantu dalam menyelesaikan Tugas Akhir ini, terutama kepada:

1. Bapak Andhika Bimantoro dan Ibu Diah Kurniawatie, orang tua penulis yang tak henti-hentinya memberikan doa, dukungan, dan semangat setiap saat.
2. Andina Diya Ayu Paramita Hapsari, saudara kandung penulis yang selalu memberikan doa dan dukungannya.
3. Bapak Dr.tech. Ir. R. V. Hari Ginardi, M.Sc. dan Bapak Dwi Sunaryono, S.Kom., M.Kom., Dosen Pembimbing penulis. Terima kasih atas semua bimbingan, tuntunan, dan arahan yang dicurahkan selama masa kuliah, terlebih selama penyusunan Tugas Akhir ini.
4. Bapak Arya Yudhi Wijaya, S.Kom., M.Kom. selaku Dosen Wali penulis, yang telah membantu dan membimbing penulis selama menempuh kuliah S1 di Teknik Informatika ITS.
5. Bapak dan Ibu Dosen Jurusan Teknik Informatika FTIf ITS, yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
6. Adella Adiningtyas yang selalu memberikan semangat dan motivasi selama penulisan Tugas Akhir ini.
7. Ibet dan Fahmi yang telah banyak memberikan bantuan dalam penyusunan Tugas Akhir ini.
8. Ibet, Melfa, Kaspul, Fandi, Fahmi, Bachmid, Ata, Kemal, Megi, Agung, teman-teman seperjuangan dan

sepermainan selama berkegiatan sehari-hari didalam dan diluar Kampus Perjuangan.

9. Keluarga Besar SAFARY Bogor, teman-teman seperantauan dan seperjuangan penulis selama berkuliah di Kota Surabaya ini.
10. Keluarga Besar Angkatan 2011 yang telah menjadi keluarga kedua penulis selama menempuh kuliah di Teknik Informatika ITS.
11. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu persatu.

Penulis telah berusaha sebaik-baiknya dalam menyusun Tugas Akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan, maupun kelalaian yang telah dilakukan. Penulis berharap Tugas Akhir ini dapat menjadi sebuah solusi yang bermanfaat bagi masyarakat, bangsa, dan negara.

Surabaya, Juni 2015

Mahardhika Rizki Bagaskoro

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

Pada bab ini dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, rumusan dan batasan permasalahan, tujuan dan manfaat pembuatan, metodologi pengerjaan, dan sistematika penulisan.

1.1 Latar Belakang

Kemajuan teknologi informasi yang pesat, berpengaruh pada perkembangan perangkat bergerak saat ini, sehingga penggunaan perangkat bergerak semakin memasyarakat. Perkembangan ini sangatlah membantu dalam menyajikan informasi yang cepat dan efisien dengan mengakses informasi melalui perangkat bergerak tersebut. Walaupun perangkat bergerak merupakan perangkat dengan layar penyajian yang sangat terbatas, tetapi penyajian informasinya pun tidak kalah optimal layaknya informasi yang diakses dari komputer *desktop*. Selain itu, kemampuan dalam memodifikasi aplikasi-aplikasi pada perangkat bergerak dengan menambahkan modul-modul yang diperlukan akan membuat perangkat bergerak dapat semakin menggeser peran komputer *desktop*. Kemudahan dan tingkat mobilitas tinggi yang ditawarkan oleh perangkat bergerak inilah yang mendorong masyarakat untuk mulai beralih menggunakan perangkat bergerak dibandingkan komputer *desktop*, tidak terkecuali elemen pemerintahan. Dengan semakin digunakannya perangkat bergerak oleh elemen pemerintahan Kota Kediri serta tingginya animo masyarakat terhadap sistem aduan Suara Warga Kota Kediri [1], maka dibutuhkan suatu aplikasi yang dapat menunjang dan mempermudah mereka dalam mengakses, memantau, dan melakukan *monitoring* melalui perangkat bergerak yang dimiliki, tanpa perlu repot untuk menggunakan komputer. Oleh karena itu, Pemerintah Kota Kediri berinisiatif untuk membuat layanan Aduan Suara Warga Kota Kediri yang sudah ada ke dalam bentuk

perangkat bergerak yang dapat diakses melalui telepon pintar petugas Pemerintah Kota Kediri.

Melakukan *monitoring* aduan dari warga tentu akan semakin mudah jika dapat dilakukan melalui aplikasi perangkat bergerak. Aduan dan tanggapan tersebut dapat dimonitoring kapan saja dan dimana saja oleh walikota ataupun petugas *monitoring.* Tanggapan yang diberikan oleh departemen tertentu terhadap aduan yang diterimanya akan mempengaruhi penilaian kinerja dari departemen yang bersangkutan. Hal seperti ini akan membuat departemen akan selalu menanggapi setiap aduan dengan sembarangan agar mendapatkan penilaian yang baik. Oleh karena itu diperlukan sebuah metode untuk mengelompokkan tanggapan yang diberikan oleh departemen tertentu terhadap aduan yang diterimananya.

Untuk mengatasi masalah tersebut, maka akan digunakan metode klasifikasi untuk melakukan pengelompokan tanggapan-tanggapan yang diberikan oleh departemen tertentu terhadap aduan yang diterimananya. Klasifikasi dalam data *mining* didefinisikan sebagai metode pembelajaran data untuk memprediksi nilai dari sekelompok atribut. Algoritma klasifikasi akan menghasilkan sekumpulan aturan yang disebut *rule* yang akan digunakan sebagai indikator untuk dapat memprediksi kelas dari data yang ingin diprediksi. Tujuan dari algoritma klasifikasi adalah untuk menemukan relasi antara beberapa variabel yang tergolong dalam kelas yang sama. Relasi tersebut akan digambarkan dengan aturan-aturan agar dapat memprediksi kelas dari data yang atributnya sudah diketahui [2]. Dalam tugas akhir ini, akan digunakan metode klasifikasi Naïve Bayes yang memungkinkan sebuah teks dikelompokkan ke dalam kelas-kelas yang berbeda.

Dengan menggunakan metode klasifikasi, tanggapan-tanggapan yang sebelumnya dianggap sama pada proses penilaian kinerja departemen dapat dipisahkan menjadi tanggapan yang baik atau tidak untuk selanjutnya mempengaruhi penilaian kinerja suatu departemen.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut.

1. Bagaimana membuat suatu modul aplikasi yang dapat diimplementasikan pada aplikasi perangkat bergerak Suara Warga Kota Kediri?
2. Bagaimana membuat modul aplikasi perangkat bergerak yang dapat memantau serta melakukan monitoring terhadap aduan suara warga Kota Kediri?
3. Bagaimana membuat modul aplikasi yang dapat melakukan pengelompokan tanggapan petugas secara otomatis menggunakan algoritma Naïve Bayes?
4. Bagaimana merepresentasikan data statistik aduan ke dalam bentuk diagram batang dan *listview* pada perangkat bergerak berbasis Android?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, diantaranya sebagai berikut.

1. Aplikasi ini adalah modul aplikasi perangkat bergerak yang berbasis pada sistem operasi Android dengan versi minimal 4.1 (Jelly Bean).
2. Modul aplikasi perangkat bergerak ini membutuhkan koneksi internet untuk beroperasi.
3. Algoritma yang digunakan dalam pengelompokan tanggapan adalah algoritma klasifikasi Naïve Bayes.
4. Proses klasifikasi pada Tugas Akhir ini dilakukan secara *offline* (*client side*) dan hanya mengklasifikasikan tanggapan tanpa memperhitungkan keterkaitan antara tanggapan dengan aduannya.
5. Jenis statistik yang ditampilkan pada modul aplikasi perangkat bergerak ini adalah statistik aduan, statistik waktu penyelesaian aduan, dan statistik rerata rating aduan, yang

ketiganya direpresentasikan ke dalam bentuk diagram batang dan *listview*.

6. Aplikasi yang dibangun merupakan modul yang mengimplementasi kelas *interface* dari aplikasi utama (Aplikasi Suara Warga Kota Kediri).

1.4 Tujuan dan Manfaat

Tujuan dari pengerjaan Tugas Akhir ini adalah sebagai berikut.

1. Untuk membuat modul aplikasi yang dapat diimplementasikan pada aplikasi perangkat bergerak Suara Warga Kota Kediri
2. Untuk membuat modul aplikasi perangkat bergerak yang dapat melakukan monitoring aduan suara warga.
3. Untuk membuat modul aplikasi perangkat bergerak yang dapat mengelompokkan tanggapan yang baik, netral atau tidak untuk kemudian dilakukan *scoring* atau penilaian kepada departemen yang bersangkutan berdasarkan hasil pengelompokan tanggapan tersebut menggunakan algoritma Naïve Bayes.
4. Untuk membuat modul aplikasi perangkat bergerak yang dapat merepresentasikan data statistik aduan ke dalam bentuk diagram batang dan *listview*.

Tugas Akhir ini dikerjakan dengan harapan dapat memberikan manfaat pada bidang teknik informatika di pemerintahan sebagai alternatif dalam melakukan *monitoring* tanggapan pengaduan warga. Tugas akhir ini juga diharapkan dapat memberikan informasi berupa kategori tanggapan yang diberikan seorang petugas pemerintahan terhadap pengaduan dari warga.

1.5 Metodologi

Ada beberapa tahap dalam proses pengerjaan Tugas Akhir ini. Berikut ini adalah tahap-tahap dalam pembuatan Tugas Akhir.

a. Studi Literatur

Tahap ini merupakan tahap pengumpulan informasi dan pembelajaran yang akan digunakan pada Tugas Akhir ini. Studi literatur meliputi diskusi dan pemahaman mengenai topik Tugas Akhir ini, diantaranya mengenai:

a. Algoritma *Naïve Bayes.*
b. Praproses pada *text mining*.

b. Desain Sistem

Tahap ini merupakan perancangan sistem dengan menggunakan studi literatur dan mempelajari konsep aplikasi yang akan dibuat. Dengan berbekal teori, metode, dan informasi yang sudah terkumpul pada tahap sebelumnya diharapkan dapat membantu dalam proses perancangan sistem.

c. Implementasi

Implementasi merupakan tahap membangun aplikasi, yaitu mengimplementasikan desain atau rancangan yang dibuat ke dalam baris kode program. Pengmbangan aplikasi ini dimulai dengan membuat fungsi pada *web service* untuk mengakses *database* pada server dengan menggunakan bahasa PHP. Kemudian dilanjutkan dengan membuat aplikasi klien Android dengan menggunakan bahasa pemrograman Java.

d. Uji Coba dan Evaluasi

Pada tahap ini dilakukan pengujian terhadap aplikasi yang dibuat, baik secara keseluruhan maupun hanya fitur klasifikasinya saja dengan menggunakan data ataupun kasus yang telah disiapkan. Tujuan pengujian ini adalah untuk menguji fungsionalitas dari

aplikasi, mencari masalah yang mungkin muncul, dan melakukan perbaikan bila ada kekurangan.

e. Penyusunan Laporan Tugas Akhir

Tahap ini digunakan untuk membuat laporan Tugas Akhir yang berisi metode, dasar teori, dan hasil yang didapatkan selama pengerjaan Tugas Akhir.

1.6 Sistematika Penyusunan Laporan

Buku Tugas Akhir ini disusun dengan sistematika laporan sebagai berikut.

1. Bab I. Pendahuluan
   Bab ini meliputi latar belakang masalah, rumusan permasalahan, batasan masalah, tujuan dan manfaat pembuatan Tugas Akhir, metodologi yang digunakan, dan sistematika penyusunan laporan Tugas Akhir.

2. Bab II. Tinjauan Pustaka
   Bab ini meliputi dasar teori dan penunjang yang berkaitan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

3. Bab III. Perancangan Perangkat Lunak
   Bab ini membahas desain dari sistem yang akan dibuat meliputi arsitektur sistem, *use case* sistem, dan perancangan antarmuka sistem.

4. Bab IV. Implementasi
   Bab ini membahas implementasi dari desain sistem yang dilakukan pada tahap desain, meliputi *pseudocode* dan implementasi antarmuka dari perangkat lunak.

5. Bab V. Uji Coba dan Evaluasi
   Bab ini membahas uji coba dari perangkat lunak yang dibuat dengan melihat keluaran yang dihasilkan oleh perangkat

lunak, analisis, dan evaluasi untuk mengetahui kemampuan perangkat lunak.

6. Bab VI. Kesimpulan dan Saran

   Bab ini berisi kesimpulan dari hasil uji coba yang dilakukan serta saran untuk pengembangan lebih lanjut perangkat lunak.

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan pengimplementasian perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap sistem yang dibuat dan berguna sebagai penunjang dalam pengembangan.

2.1 Praproses Teks Bahasa Indonesia

Pembangunan indeks dari koleksi dokumen merupakan tugas pokok pada tahapan *preprocessing* di dalam IR (*Information Retrieval*). Kualitas indeks mempengaruhi efektifitas dan efisiensi sistem IR. Indeks dokumen adalah himpuman *term* yang menunjukkan isi atau topik yang dikandung oleh dokumen.

Pembuatan indeks harus melibatkan konsep *linguistic processing* yang bertujuan untuk mengekstrak *term-term* penting dari dokumen yang direpresentasikan sebagai *bag of words*. Tiga langkah pembangunan indeks tersebut adalah sebagai berikut.

2.1.1 *Tokenization*

Tahap ini berfungsi untuk memisahkan deretan kata di dalam sebuah kalimat maupun paragraf menjadi token atau potongan kata tunggal atau *termed word*. Tahap ini juga berfungsi untuk menghilangkan karakter-karakter tertentu seperti tanda baca dan mengubah semua *token* ke bentuk huruf kecil (*lower case*) [3].

2.1.2 *Stopword Removal*

Stopword Removal adalah membuang kata-kata yang tidak memiliki banyak pengaruh dalam sebuah dokumen (*Stopword*). Kata-kata ini dimasukkan dalam daftar *stopword* atau *stoplist*, biasanya berupa preposisi, kata ganti, dan lain-lain. Jumlah *stopword* yang digunakan pada Tugas Akhir ini sebanyak 357 kata [4], dapat dilihat pada LAMPIRAN B DAFTAR STOPWORDS.

2.1.3 *Stemming*

Stemming adalah proses konversi *term* ke bentuk dasarnya. *Stemming* pada Tugas Akhir ini menggunakan algoritma *stemming* Porter [4], dimana proses kerja stemmer ini adalah sebagai berikut.

- Periksa jika kata memiliki lebih dari 2 suku kata. Jika iya, maka cek apakah kata tersebut memiliki akhiran partikel *(-kah, -lah, -pun*). Jika iya, maka hapus kata akhiran partikelnya dan kurangi jumlah suku kata sebanyak satu dari kata tersebut.
- Periksa apakah kata tersebut masih memiliki jumlah suku kata lebih dari 2 setelah di buang akhiran partikelnya. Jika iya, maka cek apakah kata tersebut memiliki kata akhiran milik (*possessive pronoun*) seperti *–ku* dan *-nya*. Jika iya, maka hapus kata akhiran milik tersebut dan kurangi jumlah suku kata sebanyak satu dari kata tersebut.
- Periksa apakah kata tersebut masih memiliki jumlah suku kata lebih dari 2 setelah di buang kata akhiran milik. Jika iya, maka cek apakah kata tersebut memiliki kata awalan pertama (*first order prefix*) seperti *meng-, meny-, men-, mem-, me-, peng-, peny-, pen-, pem-, di-, ter-, ke-*. Jika iya, maka hapus kata awalan tersebut dan kurangi jumlah suku kata sebanyak satu dari kata tersebut.
- Periksa apakah kata tersebut masih memiliki jumlah suku kata lebih dari 2 setelah di buang kata awalan pertama. Jika iya, maka cek apakah kata tersebut memiliki kata awalan kedua (*second order prefix*) seperti *ber-, be-, per-, pe-,*. Jika iya, maka hapus kata awalan tersebut dan kurangi jumlah suku kata sebanyak satu dari kata tersebut.
- Periksa apakah kata tersebut masih memiliki jumlah suku kata lebih dari 2 setelah di buang kata awalan kedua. Jika iya, maka cek apakah kata tersebut memiliki kata akhiran (*suffix*) seperti *-kan, -an, -i,*. Jika iya, maka hapus kata akhiran tersebut dan kurangi jumlah suku kata sebanyak satu dari kata tersebut.

- Kata dasar ditemukan.

Stemmer Porter untuk Bahasa Indonesia yang dikembangkan oleh Fadillah Z. Tala menghasilkan beberapa kata yang kurang dapat dipahami, seperti kata *berenang* menjadi *enang*, dan sebagainya. Hal ini disebabkan oleh ambiguitas dalam aturan morfologi Bahasa Indonesia. Namun kesalahan tersebut tidak mempengaruhi kinerja dari klasifikasi, karena pada Naïve Bayes tiap-tiap kata dalam dokumen diasumsikan tidak bergantung satu sama lain [5].

2.1.4 Contoh Praproses Tanggapan Teks Bahasa Indonesia

Misalkan ada sebuah dokumen teks atau tanggapan berbahasa Indonesia dan akan dilakukan tahap *preprocessing*

"*Terima Kasih telah menggunakan layanan pengaduan ini*".

Dari contoh di atas, apabila deilakukan tahap *preprocessing* mulai dari *tokenization* sampai *stemming*, maka hasil dari tiap-tiap proses adalah sebagai berikut.

- *Tokenization*.
  Apabila dokumen atau tanggapan Bahasa Indonesia diatas dilakukan proses tokenization, maka akan menghasilkan output sebagai berikut.

  terima – kasih – telah – menggunakan – layanan – pengaduan – ini

  Dari satu tanggapan yang berisi kalimat utuh seperti contoh di atas, setelah dilakukan proses *tokenization*, menghasilkan *output* berupa potongan masing-masing kata tunggal yang menyusun tanggapan tersebut.
- Stopword Removal.
  Dalam tahap ini dilakukan penghapusan *stopword* yang terdapat dalam tanggapan. Dari hasil proses *tokenization*

kemudian dicari semua kata yang termasuk dalam daftar *stopword*. Apabila ditemukan maka akan dihapus proses *stopword removal* adalah sebagai berikut.

terima – kasih – telah – menggunakan – layanan – pengaduan – ini

Kata yang berwarna merah termasuk kata dalam daftar *stopword*, pada proses ini kata-kata tersebut akan dihapus dan menghasilkan *output* sebagai berikut.

terima – kasih – menggunakan – layanan – pengaduan

Setelah dilakukan penghapusan *stopword*, maka menyisakan kata-kata yang memiliki pengaruh dalam suatu tanggapan.

- Stemming.
  Tahap berikutnya adalah *stemming*, dalam tahap ini setiap kata yang telah dihasilkan dari serangkaian proses sebelumnya kemudian dirubah ke bentuk dasar dari setiap kata tersebut, dan akan menghasilkan *outout* sebagai berikut.

  terima – kasih – guna – layan – adu

  Setelah proses *stemming* selesai dilakukan, maka akan didapat tanggapan yang berisi kata-kata dalam bentuk dasar seperti diatas.

Setelah tahap praproses selesai dan menghasilkan kata dasar dari semua kata yang ada dalam tanggapan, proses berikutnya yaitu menghitung frekuensi kemunculan setiap kata yang ada. Perhitungan ini dilakukan untuk menyatakan kepentingan suatu kata dalam suatu tanggapan. Metode perhitungan frekuensi yang digunakan pada Tugas Akhir ini adalah metode perhitungan *Term Frequency* (TF). TF merupakan metode yang paling sederhana

dalam melakukan pembobotan setiap *term*. Bobot dari *term* t pada tanggapan d dirumuskan sebagai berikut.

$$TF(d,t) = f(d,t) \quad \textbf{(2.1)}$$

Dimana *f(d,t)* adalah frekuensi kemunculan *term* t pada tanggapan d.

2.1.5 Contoh Perhitungan Frekuensi

Dari Persamaan (2.1) akan didapat nilai frekuensi tiap-tiap term dalam tanggapan. Dari contoh tanggapan yang telah dilakukan *preprocessing* seperti yang telah diuraikan pada subbab 2.1.1, maka didapat frekuensi kemunculan masing-masing kata dalam suatu tanggapan, seperti pada Tabel 2.1 berikut ini.

Tabel 2.1 Contoh perhitungan frekuensi

Term	TF(d,t)	
	Kemunculan	**Jumlah**
terima	1	1
kasih	1	1
guna	1	1
layan	1	1
adu	1	1

Pada contoh perhitungan frekuensi di atas diketahui bahwa semua kata hanya muncul satu kali dalam tanggapan tersebut, sehingga masing-masing kata memiliki jumlah frekuensi 1. Rangkaian tahap *preprocessing* ini dilakukan terus-menerus hingga semua tanggapan telah mengalami *preprocessing* dan menghasilkan kata-kata dalam bentuk dasar yang memiliki nilai frekuensi masing-masing.

2.2 Klasifikasi Dokumen

Klasifikasi dokumen adalah suatu proses pengelompokan dokumen sesuai dengan kategori yang dimilikinya [6]. Klasifikasi dokumen merupakan masalah yang mendasar namun sangat penting karena manfaatnya cukup besar mengingat jumlah dokumen yang ada setiap waktu semakin bertambah. Sebuah dokumen dapat dikelompokkan ke dalam kategori tertentu berdasarkan kata-kata dan kalimat-kalimat yang ada di dalam dokumen tersebut. Kata atau kalimat yang terdapat di dalam sebuah dokumen memiliki makna tertentu dan dapat digunakan sebagai dasar untuk menentukan kategori dari suatu dokumen. Dokumen yang diklasifikasikan memiliki berbagai macam jenis, seperti artikel, berita, dan lain sebagainya. Pada Tugas Akhir ini, dokumen yang akan diklasifikasikan berupa tanggapan yang diberikan oleh petugas pemerintahan Kota Kediri terhadap aduan dari warganya.

Klasifikasi termasuk pembelajaran *supervised learning*. Jenis lainnya adalah *unsupervised learning* atau dikenal sebagai *clustering*. Pada *supervised learning*, setiap data *training* mengandung pasangan data *input* dan *output* yang diharapkan, sedangkan pada *unsupervised learning* belum ditentukan target *output* yang harus diperoleh.

Proses klasifikasi teks dapat dibagi ke dalam dua fase sebagai berikut.

- Fase *information retrieval* (IR) untuk mendapatkan data numeric dari dokumen teks. Langkah pertama yang dilakukan pada fase ini adalah *feature extraction*. Pendekatan yang umum digunakan adalah distribusi frekuensi kata. Nilai numerik yang diperoleh dapat berupa berapa kali suatu kata muncul di dalam dokumen, 1 jika kata ada di dalam dokumen atau 0 jika tidak ada (biner), atau jumlah kemunculan kata pada dokumen. Fitur yang diperoleh dapat direduksi agar dimensi vektor menjadi lebih kecil. Beberapa pendekatan *feature reduction* dapat diterapkan seperti *stopword removal* dan *stemming*.

- Fase klasifikasi utama. Data numerik hasil dari proses pada fase pertama diatas akan diproses lagi untuk memutuskan ke kategori mana teks ditempatkan. Terdapat beberapa algoritma klasifikasi yang merupakan kajian di bidang statistika dan *machine learning* yang dapat diterapkan pada fase ini, diantaranya adalah *Naïve Bayes*, *Decision Tree*, *k-Nearest Neighbor* (k-NN), *Neural Network* (NN), dan *Support Vector Machines* (SVM).

Manfaat dari klasifikasi dokumen adalah untuk pengorganisasian dokumen. Dengan jumlah dokumen yang sangat besar, untuk mencari sebuah dokumen akan lebih mudah apabila kuumpulan dokumen yang dimiliki terorganisir dan telah dikelompokkan sesaui kategorinya masing-masing. Contoh aplikasi penggunaan klasifikasi dokumen teks yang banyak digunakan adalah *email spam filtering*. Pada aplikasi *spam filtering* sebuah *email* diklasifikasikan apakah *email* tersebut termasuk *spam* atau tidak dengan memperhatikan kata-kata yang terdapat dalam *email* tersebut. Aplikasi semacam ini telah banyak digunakan oleh banyak penyedia jasa layanan *email*.

2.2.1 Naïve Bayes

Naïve bayes merupakan salah satu metode *machine learning* yang menggunakan perhitungan probabilitas yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi probabilitas di masa depan berdasarkan pengalaman di masa sebelumnya. Atau dalam konsep IR metode seperti ini biasa ditandai dengan adanya satu set data yang dibagi dalam dua kelompok, data *training* dan data *testing*. Data *training* merupakan representasi pengetahuan yang akan digunakan untuk prediksi kelas data baru yang belum pernah ada, sedangkan data *testing* digunakan untuk mengukur sejauh mana *classifier* berhasil melakukan klasifikasi dengan benar. Karena itu, data yang ada pada data *testing* seharusnya tidak boleh ada pada data *training* sehingga dapat diketahui apakah model *classifier* sudah "pintar" dalam melakukan klasifikasi.

Bayesian filter atau klasifikasi Naïve Bayes merupakan metode yang digunakan untuk mengklasifikasikan sekumpulan dokumen [3]. Pada metode Naïve Bayes, sebuah dokumen teks direpresentasikan sebagai kumpulan kata-kata (*bag of words*), dimana tiap-tiap kata dalam dokumen tersebut diasumsikan tidak bergantung satu sama lain [5].

Naïve Bayes merupakan turunan dari teorema Bayes, yaitu melakukan klasifikasi dengan menghitung nilai probabilitas kategori dan semua data yang ada. Dengan pendekatan Naïve Bayes yang mengasumsikan bahwa tiap-tiap kata didalam setiap dokumen adalah tidak bergantung satu sama lain, maka perhitungan nilai probabilitas tersebut dapat dituliskan dengan menggunakan Persamaan (2.2).

$$C_{MAP} = argmax\, p(c_j) \prod_i p(w_i|c_j) \qquad \textbf{(2.2)}$$

Pada saat proses klasifikasi dokumen teks, maka pendekatan Naïve Bayes akan menyeleksi kategori teks yang memiliki probabilitas paling tinggi (C_{MAP}) seperti yang ditunjukkan pada Persamaan (2.2).

Nilai $p(c_j)$ untuk menghitung nilai probabilitas masing-masing kategori c_j dihitung menggunakan Persamaan (2.3).

$$p(c_j) = \frac{n(w_j)}{n(sampel)} \quad \textbf{(2.3)}$$

dimana:

- $n(w_j)$ adalah jumlah dokumen pada kategori j,
- n(sampel) adalah jumlah dokumen sampel yang digunakan dalam proses training.

Sementara nilai $p(w_i \mid c_j)$ untuk menghitung nilai probabilitas untuk setiap kata pada dokumen *testing* dihitung menggunakan Persamaan (2.4).

$$p(w_i|c_j) = \frac{1 + n_i}{|C| + n(kosakata)} \quad \textbf{(2.4)}$$

dimana:

- n_i adalah jumlah kemunculan kata w_i pada kategori c_j,
- $|C|$ adalah jumlah semua kata pada kategori c_j,
- n(kosakata) adalah jumlah kata yang unik pada semua data training.

Dengan Persamaan (2.3) dan Persamaan (2.4), didapatkan nilai probabilitas masing-masing kategori c_j dan nilai probabilitas untuk setiap kata pada dokumen *testing*. Selanjutnya nilai probabilitas kategori akan dikalikan dengan semua nilai probabilitas setiap kata pada dokumen *testing* dengan menggunakan Persamaan (2.2). Kemudian akan dipilih hasil perkalian terbesar yang dimana hasil tersebut merupakan kemungkinan kategori dari suatu dokumen *testing*.

2.2.2 Contoh Perhitungan Naïve Bayes

Dari Persamaan (2.3) dan Persamaan (2.4) di atas, misalkan dari 7 tanggapan berbahasa Indonesia yang terbagi atas 2 macam kategori akan didapat model probabilistik dan selanjutnya dicari nilai terbesar dari hasil perkalian masing-masing data probabilistik yang diperoleh. Sebelumnya tentu terlebih dahulu data *training* mengalami *preprocessing* dan ekstraksi fitur, hingga didapat data hasil *preprocessing* seperti pada Tabel 2.2 berikut.

Tabel 2.2 Tanggapan Setelah *Preprocessing*

Tanggapan	Kata Hasil Ekstraksi (Kemunculan)	Kategori
D_1	ganti(1), lampu(1), semampir(1), terang(1)	BAIK
D_2	untuk(1), jadi(1), pns(1), jalur(1), umum(1), formasi(1), ada(1), cpns(1), terima(1), kasih(1)	BAIK
D_3	anggar(1), pak(1), realisasi(1), selambat(1), lambat(1), desember(1), 2014(1)	BAIK
D_4	terima(1), kasih(1), atas(1), lapor(1), jl(1), lingkar(1), maskumambang(1), proses(1), baik(1)	BAIK
D_5	terima(1), kasih(1), atas(1), partisipasi(1)	TIDAK BAIK

D_6	terima(1), kasih(1), atas(1), hati(1)	TIDAK BAIK
D_7	terima(1), kasih(1), atas(1), adu(1)	?

Dari data *training* setelah tahap *preprocessing* dan ekstraksi fitur seperti pada Tabel 2.2 di atas, dengan menggunakan Persamaan (2.3) dan Persamaan (2.4), dapat dihitung nilai kemungkinan tanggapan D_7, sebagai berikut.

- Menghitung *priors* untuk setiap kategori dengan menggunakan Persamaan (**2.3)**:
  a. p(BAIK) = $\mathbf{4}/\mathbf{6}$
  b. p(TIDAK BAIK) = $2/6$

- Menghitung *conditional probabilities* untuk setiap kata pada setiap dokumen data testing terhadap masing-masing kategori dengan menggunakan Persamaan (**2.4)**:
  a. p(terima|BAIK)=$(2+1)/(30+30)=3/60$
  b. p(kasih|BAIK)=$(2+1)/(30+30)=3/60$
  c. p(atas|BAIK)=$(1+1)/(30+30)=2/60$
  d. p(adu|BAIK)=$(0+1)/(30+30)=1/60$
  e. p(terima|TIDAKBAIK)=$(2+1)/(8+30)=3/38$
  f. p(kasih|TIDAK BAIK)=$(2+1)/(8+30)=3/38$
  g. p(atas|TIDAK BAIK)= $(2+1)/(8+30)=3/38$
  h. p(adu|TIDAK BAIK)= $\mathbf{(0+1)}/\mathbf{(8+30)}=\mathbf{1}/\mathbf{38}$

- Memilih kategori untuk dokumen *testing* D_7 dengan menggunakan Persamaan (**2.2)**:

a. P(BAIK|D_7)
= p(BAIK) x p(terima|BAIK) x p(kasih|BAIK) x p(atas|BAIK) x p(adu|BAIK)
= $^4/_6 \text{x} ^3/_{60} \text{x} ^3/_{60} \text{x} ^2/_{60} \text{x} ^1/_{60}$
= $^{72}/_{77760000} \approx 9.26 \text{ x } 10^{-7}$

b. P(TIDAK BAIK|D_7)
= p(TIDAK BAIK) x p(terima| TIDAK BAIK) x p(kasih| TIDAK BAIK) x p(atas| TIDAK BAIK) x p(adu| TIDAK BAIK)
= $\mathbf{^2/_6 \text{x} ^3/_{38} \text{x} ^3/_{38} \text{x} ^3/_{38} \text{x} ^1/_{38}}$
= $\mathbf{^{54}/_{12510816} \approx 4.31 \text{ x } 10^{-5}}$

Berdasarkan hasil perhitungan Naïve Bayes di atas, diketahui bahwa D_7 menghasilkan nilai terbesar pada perhitungan dengan kategori kedua, yaitu kategori TIDAK BAIK. Maka dari itu dapat disimpulkan bahwa D_7 termasuk ke dalam kategori TIDAK BAIK. perhitungan seperti ini dilakukan pada semua dokumen *testing* sehingga semua tanggapan terklasifikasi.

2.3 Metode Evaluasi

Metode evaluasi yang diukur dalam Tugas Akhir ini adalah tingkat akurasi algoritma Naïve Bayes dalam melakukan klasifikasi tanggapan berbahasa Indonesia secara benar ke dalam tiga kategori yang telah ditentukan, yaitu Baik, Netral, dan Tidak Baik [5]. Evaluasi diukur dengan menghitung persentase kebenaran algoritma mengategorikan tanggapan dengan cara membandingkan kategori yang dihasilkan algoritma dengan data *ground truth* pada data *testing*.

Selain itu juga akan diukur tingkat akurasi menggunakan metode K-Fold Cross-Validation untuk memperkecil bias yang terkait dengan *sampling random* dari sampel data *training* dalam membandingkan akurasi prediksi [7]. Dalam K-Fold Cross-

Validation, *dataset* yang utuh di pecah secara *random* menjadi k *subset* dengan ukuran yang hampir sama dan saling eksklusif satu sama lain. Model dalam klasifikasi dilatih dan diuji sebanyak k kali. Setiap kali pelatihan dilakukan dengan 1 *subset* bergantian menjadi data *testing* sementara sisanya menjadi data *training*. Penilaian *cross-validation* terhadap akurasi model secara keseluruhan dihitung dengan mengambil rata-rata dari semua hasil akurasi individu k, seperti yang ditunjukkan dengan Persamaan (2.5).

$$CVA = \frac{i}{k}\sum_{i=1}^{k} A_i \quad \textbf{(2.5)}$$

dimana:

- CVA adalah nilai akurasi *cross-validation*,
- k adalah jumlah *fold*,
- A adalah nilai akurasi dari masing-masing *fold*.

Kemudian akan dilakukan perhitungan terhadap standar deviasi untuk mengetahui pola sebaran data yang akan memberikan gambaran mengenai karakter sampel apakah cukup konsisten untuk dapat diterima sebagai karakter sampel yang sebenarnya [8]. Perhitungan ini dilakukan dengan terlebih dahulu menghitung nilai varians menggunakan Persamaan (2.6) untuk kemudian dihitung standar deviasinya dengan menghitung akar dari varians tersebut.

$$s^2 = \frac{\sum(x - \bar{x})^2}{(n-1)} \quad \textbf{(2.6)}$$

Selain itu akan dilakukan pula Black Box *Testing*, yaitu melakukan evaluasi dan ujicoba secara menyeluruh pada antarmuka modul aplikasi dengan cara mendemonstrasikan fungsi

dari modul aplikasi dengan cara menguji apakah *input* diterima dengan benar dan *output* yang dihasilkan sudah benar [9].

2.4 Android SDK

Android SDK adalah *tools* API (*Application Programming Interface*) yang digunakan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang dirilis oleh Google. Saat ini disediakan Android SDK *(Software Development Kit)* sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. Sebagai *platform* aplikasi netral, Android memberi Anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *smartphone*. Beberapa fitur Android yang paling penting adalah [10]:

- Kakas kerja aplikasi yang mendukung penggantian komponen dan *reusable,*
- Mesin Virtual Dalvik dioptimalkan untuk perangkat bergerak,
- *Integrated browser* berdasarkan *engine open source webkit,*
- Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi openGL ES 1.0 (Opsional Akselerasi Perangkat Keras),
- SQLite untuk penyimpanan data,
- Media Support yang mendukung audio, video dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (tergantung perangakat bergerak),
- Bluetooth, EDGE, 3G, WiFi (tergantung perangakat bergerak),

- Kamera, GPS, kompas dan *accelerator* (tergantung perangakat bergerak),
- Lingkungan pengembangan yang lengkap dan kaya, termasuk perangkat emulator, *tools* untuk *debuging*, profil dan kinerja memori, dan plug in untuk IDE Eclipse.

2.5 MPAndroidChart

MPAndroidChart adalah library Android yang memungkinkan pengembang untuk dengan mudah membuat diagram pada aplikasi Android. Library ini mendukung berbagai macam jenis diagram seperti diagram garis, diagram batang, diagram lingkaran, dan lain sebagainya [11].

Library ini dikembangkan oleh Philipp Jahoda dengan lisensi Apache. Contoh aplikasi yang menggunakan MPAndroidChart diantaranya My Expenses, Phone Addiction, Drivvo.

2.6 *Web Service*

Web Service adalah mekanisme terstandar yang menghubungkan sebuah aplikasi perangkat lunak dengan aplikasi perangkat lunak lain [12]. *Web Service* menggunakan XML untuk meng-*encode* data melalui protokol HTTP sehingga hampir seluruh *platform* dapat menerima data ada dalam *web service*. Selain HTTP dan XML, *Web Service* biasanya dibangun dengan dua *platform* tambahan, yaitu WSDL dan SOAP.

WSDL *(Web Services Description Language)* merupakan dokumen XML yang mendeskripsikan operasi-operasi yang disediakan oleh *Web Service*. SOAP *(Simple Object Access Protocol)* merupakan protokol pengiriman data atau pesan berbasis XML dan HTTP *header* yang digunakan oleh *Web Service.*

Dalam Tugas Akhir ini, *Web Service* digunakan untuk menerima data dari *database* pada server serta mengirimkan data kepada aplikasi *client*.

BAB III
PERANCANGAN PERANGKAT LUNAK

Perancangan merupakan bagian penting dari pembuatan suatu perangkat lunak yang berupa perencanaan-perencanaan secara teknis aplikasi yang dibuat. Sehingga Bab ini secara khusus akan menjelaskan perancangan sistem yang dibuat dalam Tugas Akhir ini. Berawal dari deskripsi umum aplikasi hingga perancangan proses, alur, dan implementasinya.

3.1 Analisis Sistem

Modul aplikasi pada Tugas Akhir ini dibangun pada perangkat bergerak berbasis sistem operasi Android agar petugas pemerintah Kota Kediri dapat melakukan *monitoring* darimana saja dan kapan saja, serta memberi opsi alternatif bagi petugas *monitoring* untuk mengevaluasi kinerja departemen dalam menanggapi aduan warga. Oleh karena itu, pengembangan modul aplikasi ini ditujukan untuk perangkat komunikasi bergerak.

Untuk itu pada tahap analisis diperlukan untuk mendefinisikan kebutuhan yang akan dipenuhi dalam pembuatan modul aplikasi ini. Berikut penjabaran bagian-bagian tahap analisis yang mencakup deskripsi umum sistem, spesifikasi kebutuhan fungsional dan non-fungsional, serta identifikasi pengguna.

3.1.1 Deskripsi Umum Sistem

Pada Tugas Akhir ini akan dibangun suatu modul aplikasi *Monitoring* Suara Warga Kota Kediri yang berjalan pada perangkat bergerak berbasis Android. Tujuannya adalah untuk membuat petugas pemerintah Kota Kediri dapat melakukan *monitoring* darimana saja dan kapan saja, serta memberi opsi alternatif bagi petugas *monitoring* untuk mengevaluasi kinerja departemen dalam menanggapi aduan warga. Modul aplikasi ini diharapkan dapat meningkatkan dan mempermudah kegiatan *monitoring*. Untuk itu modul aplikasi ini harus dapat menangani kegiatan-kegiatan yang dilakukan pada saat *monitoring*.

Pengguna dari modul aplikasi ini adalah pihak yang bertugas untuk melakukan *monitoring* terhadap sistem pengaduan, yakni petugas yang telah diberi wewenang dengan memiliki status '1' pada atribut *role* di akun Suara Warga miliknya. Sementara jenis-jenis *role* yang ada pada sistem Suara Warga dijelaskan pada Tabel 3.1 berikut.

Tabel 3.1 Tabel Identifikasi *Role* Pengguna

Role	Nama Role	Keterangan
1	*Pool*	Akun yang dapat mengakses semua aduan dan mengarahkannya ke departemen-departemen tertentu serta melakukan *monitoring* aduan.
3	*Staff*	Akun yang menerima aduan dari *pool* dan bertugas untuk menjawab aduan tersebut.
4	*Administrator*	Akun yang dapat melakukan *setting* aplikasi, menambah *user*, menambah departemen, dan melakukan manajemen aplikasi.

Pengguna modul aplikasi ini dapat melihat statistik yang berkaitan dengan aduan yang masuk ke Pemerintah Kota Kediri, serta dapat melakukan klasifikasi terhadap tanggapan yang diberikan oleh petugas terhadap aduan yang diterima ke dalam tiga kategori, yaitu kategori Baik seperti menjawab dengan kalimat "Terima kasih atas sarannya akan segera ditindaklanjuti. Sebagai informasi Dinas Kebersihan dan Pertamanan telah mensosialisasikan kepada masyarakat untuk tidak membuang sampah sembarangan. Sosialisasi tersebut dilakukan diantaranya melalui papan yang dipasang di lokasi yang rawan menjadi tempat pembuangan sampah liar." dimana tanggapan tersebut dirasa lengkap, kemudian kategori Tidak Baik dimana tanggapan dirasa tidak lengkap dan dijawab dengan sembarangan seperti petugas hanya menjawab "Baik" atau "Oke, terima kasih", dan kategori

Netral untuk mengategorikan tanggapan yang tidak dapat diklasifikasikan ke dalam kategori Baik maupun Tidak Baik seperti menjawab dengan kata "Terima kasih telah mengadukan keluhan anda pada kami.". Setelah itu modul aplikasi ini secara otomatis akan menampilkan penilaian berdasarkan tanggapan tersebut.

Terdapat empat menu utama yang dapat diakses pengguna pada modul aplikasi ini, yaitu Statistik Aduan dimana pada menu tersebut akan ditampilkan data aduan yang diterima Pemerintah Kota Kediri ke dalam bentuk diagram batang dan daftar *listview* perdepartemen. Kemudian Statistik Waktu dimana pada menu tersebut akan ditampilkan data waktu rata-rata yang dibutuhkan suatu departemen untuk menanggapi aduan ke dalam bentuk diagram batang dan daftar *listview* perdepartemen. Kemudian Statistik Rating dimana pada menu tersebut akan ditampilkan data rating rata-rata yang diberikan warga untuk suatu departemen ke dalam bentuk diagram batang dan daftar *listview* perdepartemen. Selanjutnya menu Tanggapan Departemen dimana pada menu tersebut terdapat fitur klasifikasi yang akan mengklasifikasikan tanggapan yang diberikan departemen terhadap suatu aduan, kemudian secara otomatis akan memberi penilaian kepada departemen yang direpresentasikan ke dalam bentuk diagram batang dan diagram lingkaran.

Untuk membangun modul aplikasi pada perangkat bergerak yang terhubung dengan sistem yang sudah ada, digunakan *web service* untuk dapat berkomunikasi ke *database* yang ada pada *server* seperti yang ada pada situs *web*. Perangkat bergerak berkomunikasi dengan *web service*. *Web service* berkomunikasi dengan *server* dan melanjutkannya ke perangkat bergerak.

3.1.2 Spesifikasi Kebutuhan Fungsional

Berdasarkan deskripsi umum sistem sistem diatas, maka dapat disimpulkan bahwa kebutuhan fungsionalitas dari modul aplikasi ini adalah sebagai berikut.

1. Menampilkan statistik aduan keseluruhan dalam bentuk diagram batang.

2. Menampilkan statistik aduan perbulan setiap departemen dalam bentuk diagram batang.
3. Menampilkan statistik aduan setiap departemen dalam bentuk daftar *listview.*
4. Menampilkan statistik rata-rata waktu pelayanan setiap departemen dalam bentuk diagram batang.
5. Menampilkan statistik rata-rata waktu pelayanan setiap departemen dalam bentuk daftar *listview.*
6. Menampilkan statistik rata-rata rating setiap departemen dalam bentuk diagram batang.
7. Menampilkan statistik rata-rata rating setiap departemen dalam bentuk daftar *listview.*
8. Menampilkan tanggapan-tanggapan yang diberikan oleh setiap departemen.
9. Menampilkan *score* semua departemen dalam bentuk diagram batang.
10. Menampilkan *score* detail setiap departemen dalam bentuk diagram batang.
11. Menampilkan *score* detail setiap departemen dalam bentuk diagram lingkaran.
12. Mengklasifikasi tanggapan pada suatu departemen.
13. Menampilkan daftar departemen.
14. Mencari departemen.
15. Mengurutkan departemen.

3.1.3 Spesifikasi Kebutuhan Non-Fungsional

Berikut daftar kebutuhan non-fungsional yang harus dipenuhi agar modul aplikasi berjalan sesuai kebutuhan:

1. Keamanan
   Sebagai otentikasi petugas yang mengakses sistem.
2. Koneksi internet
   Koneksi internet dibutuhkan untuk dapat mengakses *server.*

3.1.4 Identifikasi Pengguna

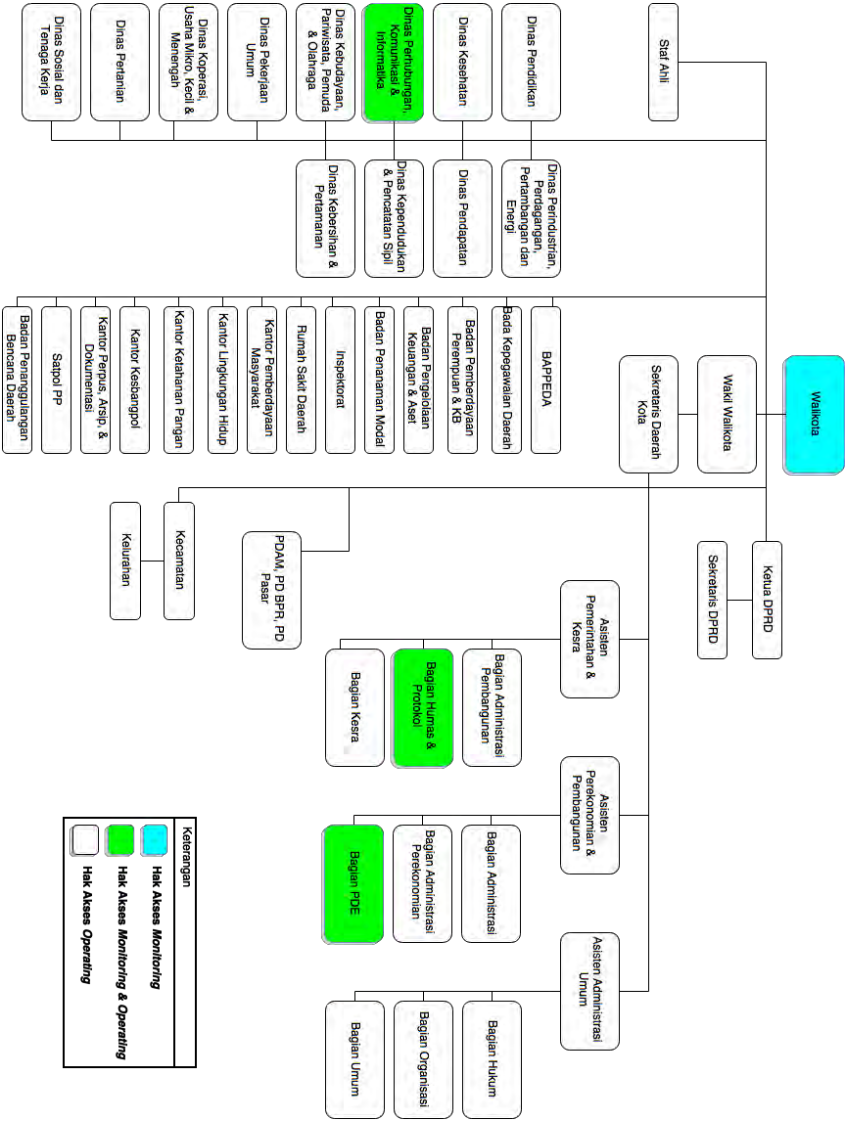
Berdasarkan deskripsi umum diatas, maka dapat diketahui bahwa pengguna yang akan menggunakan modul aplikasi ini adalah petugas atau pengguna yang memiliki hak melakukan monitoring pada Sistem Suara Warga Kota Kediri, diantaranya adalah Walikota, Bagian Humas & Protokol, Bagian PDE, Dinas Perhubungan, Komunikasi, dan Informatika, serta petugas-petugas yang diberi kewenangan oleh walikota di masa mendatang. Petugas-petugas tersebut ditandai dengan nilai 1 pada atribut *role* pada akun sistem Suara Warga Kota Kediri miliknya. Tabel 3.2 menjelaskan wewenang yang dimiliki oleh pengguna dari modul aplikasi ini dimana wewenang *monitoring* adalah wewenang untuk melakukan *monitoring* suara warga menggunakan modul aplikasi ini, sedangkan *operating* adalah wewenang untuk menggunakan aplikasi utama Suara Warga Kota Kediri.

Tabel 3.2 Tabel Identifikasi Pengguna Modul Aplikasi *Monitoring* Suara Warga

Pengguna	Wewenang	Keterangan
Walikota	*Monitoring*	Kepala daerah Kota Kediri
Bagian Humas & Protokol	*Monitoring* & *Operating*	Petugas *superuser*
Bagian PDE	*Monitoring* & *Operating*	Petugas *superuser*
Dinas Perhubungan, Komunikasi & Informatika	*Monitoring* & *Operating*	Petugas *superuser*

Gambar 3.1 menggambarkan struktur organisasi pengguna dari modul aplikasi *Monitoring* Suara Warga Kota Kediri ini. Dimana warna biru menunjukkan pengguna yang memiliki hak akses *monitoring*, warna putih menunjukkan pengguna yang hanya memiliki hak akses *operating*, sementara warna hijau menunjukkan

pengguna *superuser* yang dapat mengakses *monitoring* dan *operating*. Pengguna *superuser* ini yang memiliki kemampuan untuk memberikan hak akses *monitoring* kepada petugas lain yang diberi wewenang oleh Walikota di masa mendatang.



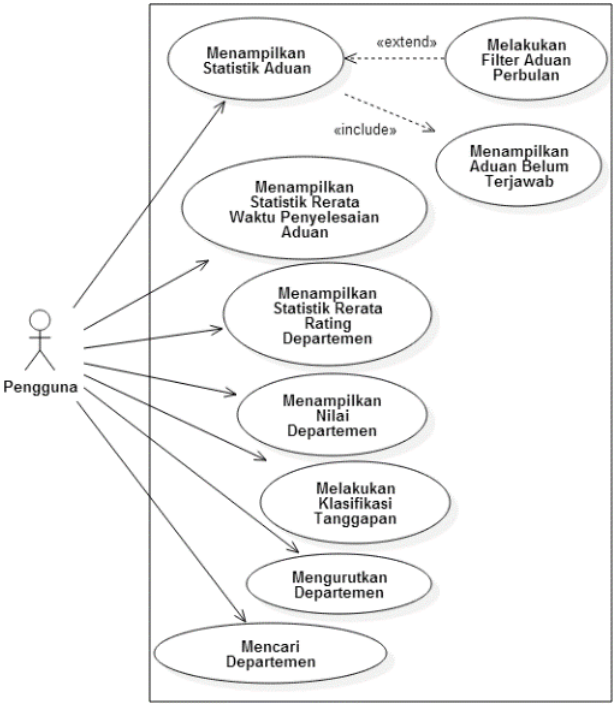
Gambar 3.1 Struktur Organisasi Pengguna Modul Aplikasi

3.2 Perancangan Sistem

Tahap perancangan dalam subbab ini dibagi menjadi beberapa bagian, yaitu perancangan *use case* sistem, arsitektur sistem, data, proses klasifikasi tanggapan, proses modularitas, dan antarmuka sistem.

3.2.1 Perancangan *Use Case* Sistem

Use case sistem merupakan diagram kebutuhan yang menggambarkan fungsionalitas sistem dan aktor-aktornya. Pada modul aplikasi ini hanya ada satu aktor yang terlibat yaitu pengguna. Pengguna berinteraksi dengan *client* untuk menjalankan segala fungsionalitas sistem. *Use case* pada sistem secara umum digambarkan pada Gambar 3.2 dan dijelaskan pada Tabel 3.3.



Gambar 3.2 Diagram *Use Case*

Tabel 3.3 Tabel Deskripsi *Use Case*

No.	Kode	Nama	Keterangan
1.	UC-01	Menampilkan statistik aduan	Pengguna dapat melihat statistik aduan yang masuk dalam bentuk diagram batang dan daftar *listview.*
2.	UC-02	Melakukan filter aduan perbulan	Pengguna dapat melihat statistik aduan yang masuk pada bulan tertentu dalam bentuk diagram batang dan daftar *listview.*
3.	UC-03	Menampilkan statistik rerata waktu penyelesaian aduan	Pengguna dapat melihat statistik rata-rata waktu yang dibutuhkan suatu departemen untuk menanggapi aduan dalam bentuk diagram batang dan daftar *listview*.
4.	UC-04	Menampilkan statistik rerata rating departemen	Pengguna dapat melihat statistik rata-rata rating suatu departemen yang diberikan warga dalam bentuk diagram batang dan daftar *listview*.
5.	UC-05	Menampilkan aduan belum terjawab	Pengguna dapat melihat aduan mana saja yang belum dijawab oleh suatu departemen.
6.	UC-06	Menampilkan nilai departemen	Pengguna dapat melihat nilai departemen yang ada dalam bentuk dalam bentuk diagram batang dan diagram lingkaran.

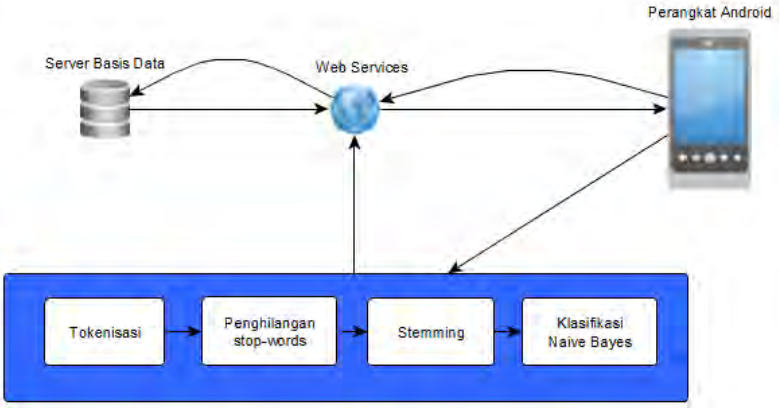
7.	UC-07	Melakukan klasifikasi Tanggapan	Pengguna dapat melakukan klasifikasi terhadap tanggapan yang telah diberikan oleh suatu departemen.
8.	UC-08	Mengurutkan departemen	Pengguna dapat mengurutkan departemen pada daftar *listview* departemen berdasarkan berbagai macam parameter seperti nama departemen, id departemen, dan lain sebagainya.
9.	UC-09	Mencari departemen	Pengguna dapat mencari departemen pada daftar *listview* departemen dengan berbagai macam kata kunci seperti nama departemen, id departemen, dan lain sebagainya.

3.2.2 Perancangan Arsitektur Sistem

Modul aplikasi ini dirancang memiliki dua sisi, yaitu sisi *server* dan sisi *client*. *Server* merupakan *web service* dan *database server*, sedangkan *client* merupakan modul aplikasi yang beroperasi pada perangkat bergerak berbasis Android. *Server* bertugas untuk menerima data dari *client*, kemudian mengolahnya dan mengirimkan hasilnya kepada *client*. *Client* bertugas menerima data dari *server* dan mengolahnya untuk kemudian menampilkannya kepada pengguna. Selain melakukan proses dan menampilkan klasifikasi, *client* juga akan menyimpan hasil klasifikasi ke dalam *database*.

Berdasarkan perancangan arsitektur umum sistem pada Gambar 3.3, data-data yang akan ditampilkan pada diagram batang dan daftar *listview* dikirim oleh *server* dengan cara *server* melakukan *query* ke *database* yang kemudian hasil dari *query*

tersebut dikirimkan ke *client* untuk diproses dan ditampilkan oleh *client*. Untuk proses klasifikasi, *client* akan melakukan proses klasifikasi terhadap tanggapan yang didapat dari *database server*, yang kemudian hasilnya akan kembali disimpan ke *database server* melalui *web service*.



Gambar 3.3 Perancangan Arsitektur Umum Sistem

3.2.3 Perancangan Data

Perancangan data adalah hal yang penting dalam sistem, karena diperlukan data yang tepat agar sistem dapat beroperasi dengan benar. Sistem yang dibuat membutuhkan data masukan dan akan memberikan data keluaran. Data masukan berupa himpunan data yang akan digunakan sebagai data latih. Data keluaran dari sistem ini adalah hasil klasifikasi tanggapan dan penilaian untuk departmen.

3.2.3.1 Data Pelatihan Untuk Klasifikasi Tanggapan

- **Data masukan**

Himpunan data yang digunakan sebagai data masukan adalah tanggapan yang telah dikumpulkan ke dalam sebuah file csv (*Comma Separated Value*). Tanggapan-tanggapan yang dihimpun sebagai data pelatihan berasal dari tanggapan yang diberikan oleh petugas Pemerintah Kota Kediri pada bulan Oktober 2014. Kelas

dalam himpunan data adalah 3 macam kategori yang mengelompokkan tanggapan-tanggapan tersebut, yaitu Baik, Netral, dan Tidak Baik. Sedangkan atribut dalam himpunan data adalah isi dari tanggapan.

- **Data keluaran**

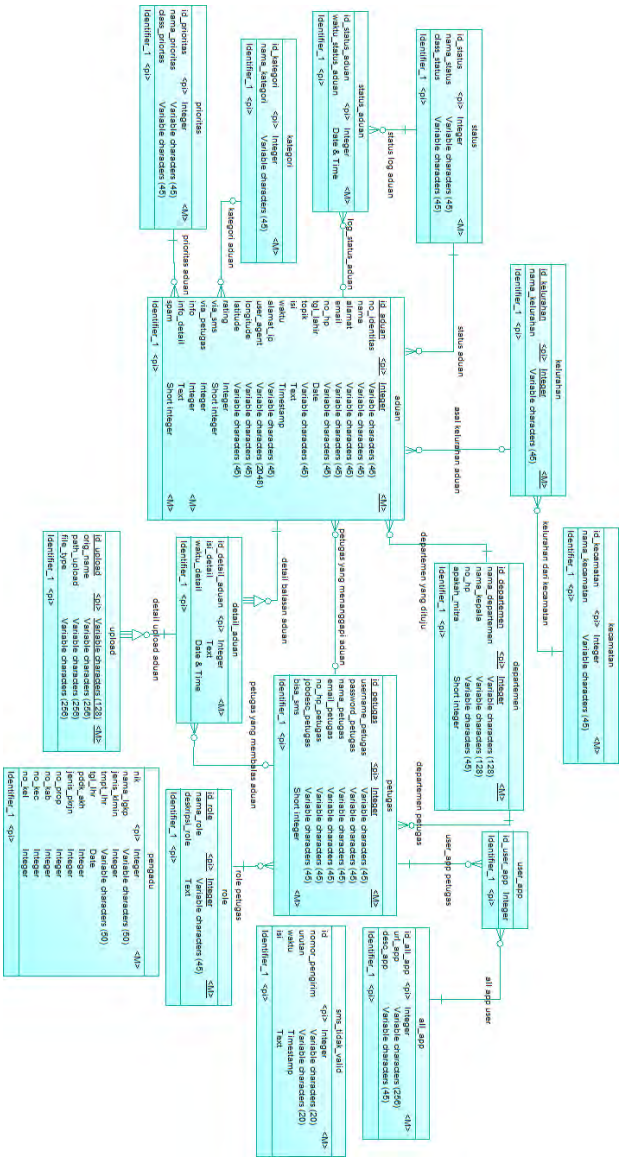
  Data keluaran dari proses pelatihan untuk klasifikasi tanggapan pada sistem ini adalah berupa kelas dari tanggapan tersebut serta nilai untuk setiap tanggapan yang akan digunakan untuk penilaian departemen. Hasil klasifikasi dan nilai tanggapan akan disimpan ke dalam sebuah tabel di dalam *database*.

3.2.3.2 Database Sistem

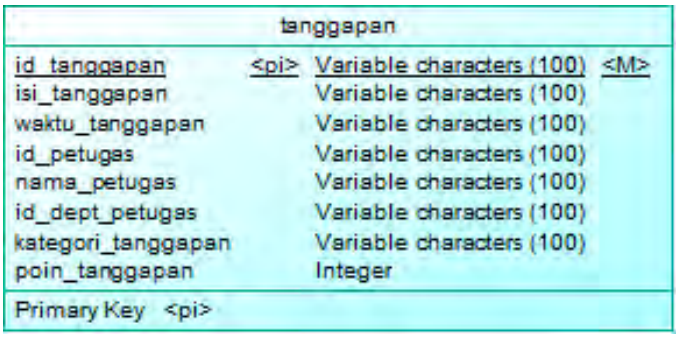
Pada subbab ini dijelaskan tentang rancangan *database* yang akan digunakan pada modul aplikasi. *Database* yang ada pada sistem ini menggunakan RDBMS MySQL.

Database digunakan untuk menyimpan informasi yang dibutuhkan baik dalam modul aplikasi ini maupun situs *web* Suara Warga Kota Kediri yang sudah ada. Database yang digunakan pada modul aplikasi ini sebanyak dua buah, database pertama adalah database yang telah ada pada server Suara Warga Kota Kediri. Database kedua adalah database yang dibuat untuk memenuhi kebutuhan penyimpanan informasi pada fitur klasifikasi pada modul aplikasi ini.

Conceptual Data Model (CDM) adalah diagram yang menjelaskan desain *database* secara *independent* tanpa mengacu pada spesifikasi *database* tertentu. CDM berfungsi untuk menggambarkan tabel yang digunakan beserta relasinya Gambar 3.4 menunjukkan diagram *Conceptual Data Model* (CDM) untuk *database* pertama yang merupakan *database* Suara Warga Kota Kediri, sedangkan Gambar 3.5 menunjukkan diagram *Conceptual Data Model* (CDM) untuk database kedua yang berfungsi untuk menyimpan informasi pada fitur klasifikasi pada modul aplikasi ini. Penjelasan terperinci mengenai tabel yang ada pada *database* untuk Tugas Akhir ini dijelaskan dalam LAMPIRAN A PERANCANGAN DATA.



Gambar 3.4 Diagram CDM *Database* Suara Warga Kota Kediri



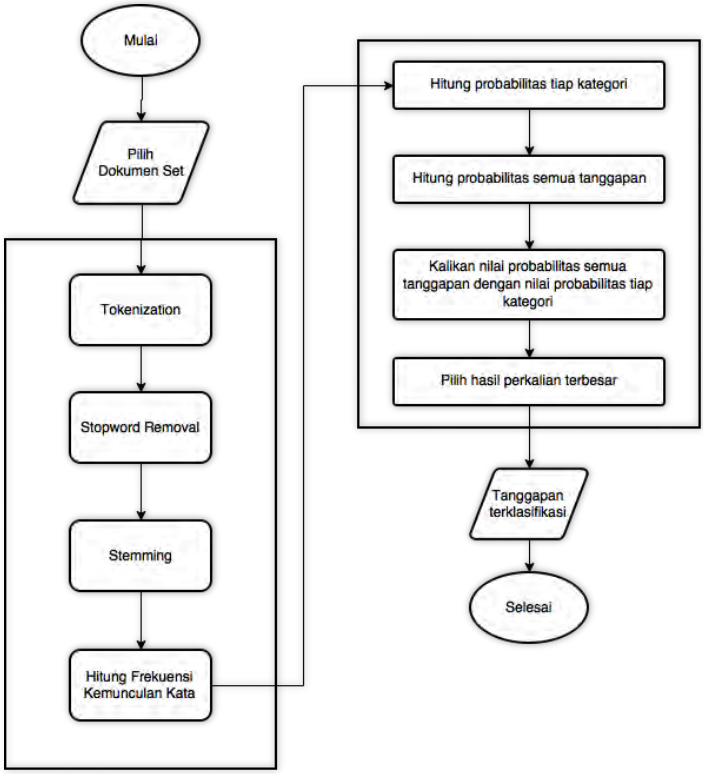
Gambar 3.5 Diagram CDM *Database* Klasifikasi

3.2.4 Perancangan Proses Klasifikasi Tanggapan

Secara garis besar, skema klasifikasi yang dibangun dalam Tugas Akhir ini terdiri dari dua tahap, yaitu tahap praproses tanggapan dan tahap klasifikasi tanggapan.

Pada tahap praproses terdapat beberapa proses yang saling berkesinambungan sehingga dapat menjadi satu kesatuan tahap praproses. Beberapa proses dalam tahap praproses itu diantaranya: *tokenization*, *stopword removal*, *stemming*, dan *term* frekuensi.

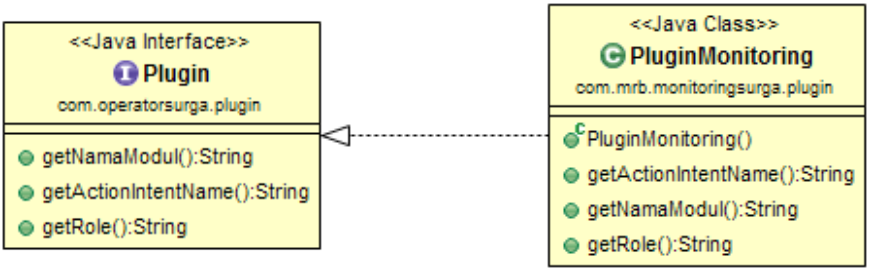
Diagram alir seluruh proses dalam klasifikasi tanggapan pada Tugas Akhir ini terlihat seperti pada Gambar 3.6. Pada diagram alir sebelah kiri menunjukkan tahap praproses tanggapan, sedangkan pada diagram alir sebelah kanan menunjukkan proses klasifikasi tanggapan menggunakan algoritma Naïve Bayes.



Gambar 3.6 Diagram Alir Perancangan Proses Klasifikasi Tanggapan

3.2.5 Perancangan Proses Modularitas

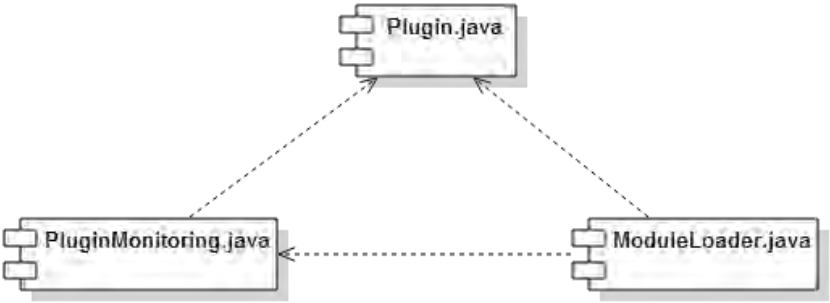
Pada Tugas Akhir ini, proses modularitas berfungsi agar modul aplikasi ini dapat diimplementasikan pada aplikasi utama. Secara garis besar, hal ini dilakukan dengan cara membuat kelas yang akan mengimplementasikan kelas *interface* milik aplikasi utama dengan terlebih dahulu meng-*import library* dari aplikasi utama. Diagram kelas untuk kelas *interface* tersebut ditunjukkan pada Gambar 3.7.



Gambar 3.7 Diagram Kelas dari *Interface* Aplikasi Utama

Kelas *interface* milik aplikasi utama memiliki tiga fungsi, yaitu *getActionIntentName()* yang berfungsi untuk memberi informasi berupa nama *package* dari modul aplikasi yang mengimplementasikan kelas *interface* aplikasi utama, fungsi *getNamaModul()* yang berfungsi untuk memberi informasi berupa nama modul dari modul aplikasi yang mengimplementasikan kelas *interface* aplikasi utama, dan fungsi *getRole()* yang berfungsi untuk memberi informasi berupa jenis role petugas yang dapat mengakses modul aplikasi ini.

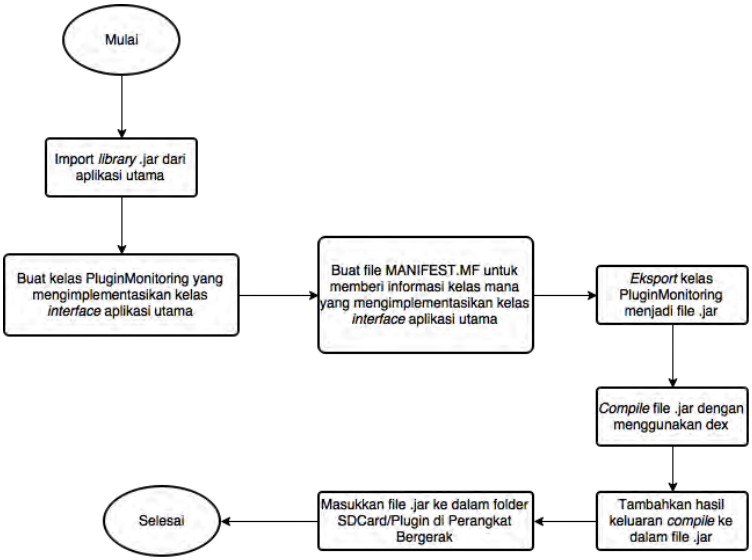
Sedangkan untuk hubungan antara komponen modularitas pada aplikasi utama dan modul aplikasi ini dapat dilihat pada Gambar 3.8.



Gambar 3.8 Diagram Komponen Modularitas

Komponen PluginMonitoring.java adalah kelas pada modul aplikasi ini yang mengimplementasikan komponen Plugin.java milik aplikasi utama. Lalu komponen PluginMonitoring.java dan Plugin.java akan dipakai oleh komponen ModuleLoader.java untuk mendapatkan informasi-informasi dari modul ini agar dapat menjalankan modul aplikasi ini.

Sementara diagram alir untuk seluruh proses modularitas pada Tugas Akhir ini terlihat seperti pada Gambar 3.9.



Gambar 3.9 Diagram Alir Perancangan Proses Modularitas

3.2.6 Perancangan Antarmuka Sistem

Pada Tugas Akhir ini, antarmuka sistem hanya ada pada modul aplikasi *client*. Rancangan antarmuka sistem meliputi, antarmuka navigasi, antarmuka statistik aduan, antarmuka statistik waktu, antarmuka statistik rating, antarmuka tanggapan departemen, antarmuka *scoring,* dan antarmuka *sort*.

3.2.6.1 Perancangan Antarmuka Navigasi

Perancangan antarmuka navigasi ditunjukkan pada Gambar 3.10. Antarmuka ini berfungsi sebagai kontrol untuk pengguna dalam mengakses menu-menu yang ada pada modul aplikasi ini, yaitu menu statistik aduan, statistik waktu, statistik rating, dan tanggapan departemen. Antarmuka navigasi diakses dengan cara menarik jari dari sisi kiri layar perangkat bergerak pengguna.

Gambar 3.10. Perancangan Antarmuka Navigasi

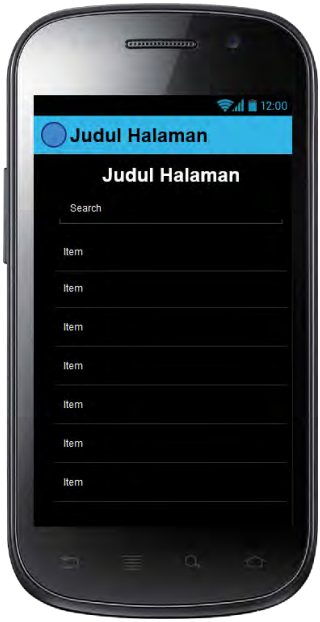
3.2.6.2 Perancangan Antarmuka Statisik Aduan

Antarmuka statistik aduan merupakan antarmuka pertama yang muncul ketika aplikasi dibuka. Pada antarmuka ini terdapat dua halaman utama yang akan ditampilkan. Kedua halaman tersebut adalah halaman diagram seperti yang ditunjukkan pada Gambar 3.11 dan halaman list seperti yang ditunjukkan pada Gambar 3.12.

Gambar 3.11. Perancangan Antarmuka Halaman Diagram Pada Menu Statistik Aduan

Komponen-komponen yang ada Gambar 3.11 adalah sebagai berikut.

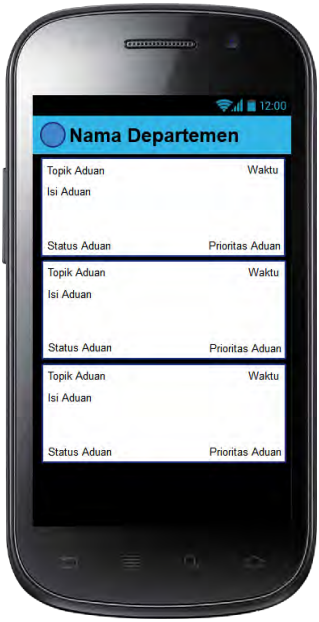
a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. Button Diagram berfungsi untuk mengubah halaman view menjadi halaman diagram.
c. Button List berfungsi untuk mengubah halaman view menjadi halaman list seperti pada Gambar 3.12.
d. Layout Diagram berfungsi sebagai tempat diagram batang akan ditampilkan pada menu statistik aduan ini.



Gambar 3.12 Perancangan Antarmuka Halaman List Pada Menu Statistik Aduan

Komponen-komponen yang ada Gambar 3.12 adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. TextBox Search berfungsi untuk mencari departemen dengan kata kunci yang dimasukkan pengguna.
c. List berfungsi untuk menampilkan daftar departemen beserta jumlah aduan yang masuk untuk setiap departemen. Dimana jika setiap item pada list dipilih, maka akan menampilkan antarmuka baru yang berisi aduan-aduan mana saja yang belum dijawab oleh departemen terkait seperti yang ditunjukkan pada Gambar 3.13.



Gambar 3.13 Perancangan Antarmuka Aduan Belum Terjawab

Komponen-komponen yang ada Gambar 3.13 adalah sebagai berikut.

a. TextView Topik Aduan digunakan untuk menampilkan topik aduan yang belum terjawab.
b. TextView Isi Aduan digunakan untuk menampilkan isi aduan yang belum terjawab.
c. TextView Status Aduan digunakan untuk menampilkan status aduan yang belum terjawab.
d. TextView Prioritas Aduan digunakan untuk menampilkan prioritas aduan yang belum terjawab
e. TextView Waktu digunakan untuk menampilkan waktu aduan tersebut diterima.

3.2.6.3 Perancangan Antarmuka Statistik Waktu

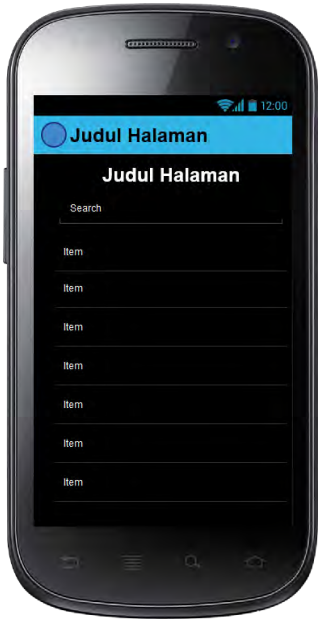
Antarmuka statistik waktu merupakan antarmuka yang muncul ketika menu statistik waktu dipilih melalui antarmuka navigasi. Pada antarmuka ini terdapat dua halaman utama yang akan ditampilkan. Kedua halaman tersebut adalah halaman diagram seperti yang ditunjukkan pada Gambar 3.14 dan halaman list seperti yang ditunjukkan pada Gambar 3.15.

Gambar 3.14 Perancangan Antarmuka Halaman Diagram Pada Menu Statistik Waktu

Komponen-komponen yang ada Gambar 3.14 adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. Button Diagram berfungsi untuk mengubah halaman view menjadi halaman diagram.

c. Button List berfungsi untuk mengubah halaman view menjadi halaman list seperti pada Gambar 3.15.
d. Layout Diagram berfungsi sebagai tempat diagram batang akan ditampilkan pada menu statistik waktu ini.



Gambar 3.15 Perancangan Antarmuka Halaman List Pada Menu Statistik Waktu

Komponen-komponen yang ada Gambar 3.15 adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. TextBox Search berfungsi untuk mencari departemen dengan kata kunci yang dimasukkan pengguna.
c. List berfungsi untuk menampilkan daftar departemen beserta waktu rata-rata penyelesaian aduan untuk setiap departemen.

3.2.6.4 Perancangan Antarmuka Statistik Rating

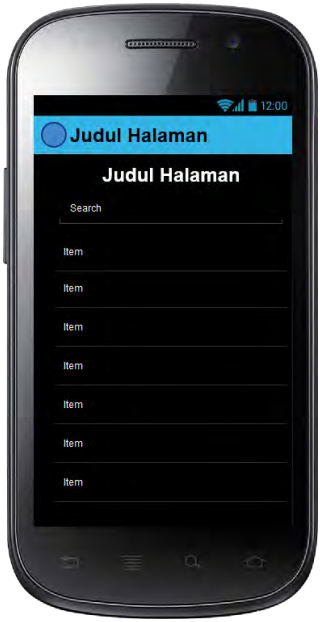
Antarmuka statistik rating merupakan antarmuka yang muncul ketika menu statistik rating dipilih melalui antarmuka navigasi. Pada antarmuka ini terdapat dua halaman utama yang akan ditampilkan. Kedua halaman tersebut adalah halaman diagram seperti yang ditunjukkan pada Gambar 3.16 dan halaman list seperti yang ditunjukkan pada Gambar 3.17.

Gambar 3.16 Perancangan Antarmuka Halaman Diagram Pada Menu Statistik Rating

Komponen-komponen yang ada Gambar 3.16 adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. Button Diagram berfungsi untuk mengubah halaman view menjadi halaman diagram.

c. Button List berfungsi untuk mengubah halaman view menjadi halaman list seperti pada Gambar 3.17.
d. Layout Diagram berfungsi sebagai tempat diagram batang akan ditampilkan pada menu statistik rating ini.



Gambar 3.17 Perancangan Antarmuka Halaman List Pada Menu Statistik Rating

Komponen-komponen yang ada Gambar 3.17 adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. TextBox Search berfungsi untuk mencari departemen dengan kata kunci yang dimasukkan pengguna.
c. List berfungsi untuk menampilkan daftar departemen beserta nilai rata-rata rating untuk setiap departemen.

3.2.6.5 Perancangan Antarmuka *Tanggapan Departemen*

Antarmuka tanggapan departemen merupakan antarmuka yang muncul ketika menu tanggapan departemen dipilih melalui antarmuka navigasi. Pada antarmuka ini terdapat list seperti yang ditunjukkan pada Gambar 3.18.

Gambar 3.18 Perancangan Antarmuka Halaman Tanggapan Departemen

Komponen-komponen yang ada Gambar 3.18adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. TextBox Search berfungsi untuk mencari departemen dengan kata kunci yang dimasukkan pengguna.
c. List berfungsi untuk menampilkan daftar departemen beserta jumlah tanggapan yang sudah ditanggapi oleh setiap departemen. Dimana jika setiap item pada list dipilih, maka

akan menampilkan antarmuka baru yang berisi tanggapan-tanggapan yang telah dibuat oleh departemen terkait seperti yang ditunjukkan pada Gambar 3.19. Dimana pada antarmuka tersebut, pengguna dapat melakukan klasifikasi tanggapan baik secara satu persatu maupun seluruh tanggapan pada suatu departemen sekaligus.

Gambar 3.19 Perancangan Antarmuka Aduan Belum Terjawab

Komponen-komponen yang ada Gambar 3.19 adalah sebagai berikut.

a. TextView ID Tanggapan digunakan untuk menampilkan id dari tanggapan.
b. TextView Isi Tanggapan digunakan untuk menampilkan isi dari tanggapan.
c. TextView Nama Petugas digunakan untuk menampilkan nama petugas yang membuat suatu tanggapan.

d. TextView Hasil Klasifikasi digunakan untuk menampilkan kategori klasifikasi dari suatu tanggapan.
e. TextView Waktu digunakan untuk menampilkan waktu tanggapan tersebut dibuat.

3.2.6.6 Perancangan Antarmuka *Scoring*

Antarmuka ini digunakan untuk melihat nilai setiap departemen dalam bentuk diagram batang. Pada antarmuka ini terdapat sebuah *layout* untuk diagram batang dan sebuah tombol seperti yang ditunjukkan pada Gambar 3.20.

Gambar 3.20 Perancangan Antarmuka *Scoring* (1)

Komponen-komponen yang ada Gambar 3.20 adalah sebagai berikut.

a. Layout Diagram berfungsi sebagai tempat diagram batang akan ditampilkan pada antarmuka tanggapan departemen ini.

b. *Button Score* Perdepartemen digunakan untuk melihat daftar departemen seperti yang ditunjukkan pada Gambar 3.21.

Gambar 3.21 Perancangan Antarmuka *Scoring* (2)

Komponen-komponen yang ada Gambar 3.21 adalah sebagai berikut.

a. TextView Judul Halaman digunakan untuk menampilkan judul menu.
b. TextBox Search berfungsi untuk mencari departemen dengan kata kunci yang dimasukkan pengguna.
c. List berfungsi untuk menampilkan daftar departemen. Dimana jika setiap item pada list dipilih, maka akan menampilkan antarmuka baru yang berisi *score* dari departemen yang dipilih seperti yang ditunjukkan pada Gambar 3.22.

Gambar 3.22 Perancangan Antarmuka *Scoring* Perdepartemen

Komponen-komponen yang ada Gambar 3.22 adalah sebagai berikut.

a. *Button* Ubah Diagram digunakan untuk mengubah diagram yang ditampilkan, dari diagram batang menjadi diagram lingkaran dan sebaliknya.
b. Layout Diagram berfungsi sebagai tempat diagram batang atau diagram lingkaran akan ditampilkan pada antarmuka *scoring* perdepartemen ini.

3.2.6.7 Perancangan Antarmuka *Sort*

Antarmuka ini digunakan untuk memilih parameter pengurutan yang diinginkan pengguna dalam mengurutkan daftar departemen pada *listview*. Antarmuka ini muncul berupa *dialog box* seperti yang ditunjukkan pada Gambar 3.23.

Gambar 3.23 Perancangan Antarmuka *Sort*

Komponen-komponen yang ada Gambar 3.23 adalah sebagai berikut.

a. *TextView Sort By* merupakan *header* atau judul dari *dialog box.*
b. *List item* berfungsi untuk menampilkan opsi parameter yang tersedia untuk mengurutkan departemen.

BAB IV
IMPLEMENTASI PERANGKAT LUNAK

Proses implementasi sistem ini dilakukan setelah melewati proses analisis dan perancangan perangkat lunak. Dalam Bab ini akan dibahas mengenai algoritma, *pseudocode,* diagram alir serta implementasi dari perancangan antarmuka yang terdapat dalam perangkat lunak sebagaimana telah dibahas pada Bab III.

4.1 Lingkungan Implementasi

Dalam merancang dan mengimplementasikan perangkat lunak ini, digunakan beberapa perangkat pendukung sebagai berikut.

4.1.1 Lingkungan Implementasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan pada lingkungan pengembangan perangkat lunak ini adalah sebagai berikut.

- Komputer Laptop Lenovo Y510P
  - Windows 8.1 Pro 64-bit,
  - Prosesor Intel® Core(TM) i7-4700MQ CPU @ 2.40GHz, dan
  - RAM 8.00 GB.
- Perangkat Bergerak Samsung Galaxy Note 3 GT-N900
  - Sistem Operasi: Android v5.0 (Lollipop),
  - CPU: 1.9 GHz Exynos
  - Memory internal: 32 GB,
  - RAM: 3.00 GB,
  - Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps, dan
  - WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot

4.1.2 Lingkungan Implementasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pengembangan sistem adalah sebagai berikut.

- Microsoft Windows 8.1 Pro sebagai sistem operasi,
- Eclipse Juno v4.2.1 sebagai IDE untuk mengimplementasikan modul aplikasi *mobile*.
- MySQL sebagai *database server,*
- StarUML dan draw.io untuk merancang diagram sistem.

4.2 Implementasi Modul *Monitoring* Suara Warga

Pada tugas akhir ini, *monitoring* merupakan modul yang memungkinkan pengguna untuk melakukan *monitoring* terhadap sistem Suara Warga Kota Kediri. Pada modul ini akan ditampilkan diagram-diagram batang untuk mendukung proses *monitoring* tersebut. Selain itu pada modul ini, diimplementasikan proses klasifikasi yang berfungsi untuk memberi nilai suatu departemen berdasarkan hasil klasifikasi dari tanggapan yang diberikannya. Implementasi pada modul ini akan dijelaskan pada subbab-subbab berikut ini.

4.2.1 Implementasi Modularitas

Pada Tugas Akhir ini, sistem yang dibangun berupa modul aplikasi yang dapat diimplementasikan ke aplikasi utama dengan cara mengimplementasikan kelas *interface* milik aplikasi utama. Kelas interface tersebut diimplementasikan pada modul aplikasi ini melalui sebuah kelas yang dibuat pada modul aplikasi ini. Implementasi kelas tersebut dapat dilihat seperti pada potongan Kode Sumber 4.1 berikut.

```
 1  import kelas interface aplikasi utama
 2  getActionIntentName() {
 3      return nama package modul
 4  }
 5  getNamaModul() {
 6      return nama modul
 7  }
 8  getRole() {
 9      return role petugas
10  }
```

Kode Sumber 4.1 *Pseudocode* Implementasi Modularitas

Pada Kode Sumber 4.1 diatas, terdapat 3 buah fungsi aplikasi utama yang diimplementasi oleh modul aplikasi ini, yaitu fungsi *getActionIntentName()* yang berfungsi untuk memberi informasi berupa nama *package* dari modul aplikasi yang mengimplementasikan kelas *interface* aplikasi utama, fungsi *getNamaModul()* yang berfungsi untuk memberi informasi berupa nama modul dari modul aplikasi yang mengimplementasikan kelas *interface* aplikasi utama, dan fungsi *getRole()* yang berfungsi untuk memberi informasi berupa jenis role petugas yang dapat mengakses modul aplikasi ini.

Selain kelas PluginMonitoring diatas, diperlukan sebuah *file* MANIFEST.MF yang berfungsi untuk memberikan informasi kepada aplikasi utama bahwa kelas PluginMonitoring pada modul aplikasi ini merupakan kelas yang mengimplementasikan kelas *interface* pada aplikasi utama. Implementasi *file* tersebut dapat dilihat seperti pada potongan Kode Sumber 4.2 berikut.

```
1  Manifest-Version: 1.0
2  Module-Class:
3  com.mrb.monitoringsurga.plugin.PluginMonitoring
```

Kode Sumber 4.2 Implementasi *File* MANIFEST.MF

Pada Kode Sumber 4.2 diatas, atribut Module-Class akan memberikan informasi kepada aplikasi utama bahwa kelas PluginMonitoring pada *package* com.mrb.monitoringsurga merupakan kelas yang mengimplementasikan kelas *interface* pada aplikasi utama.

Selain kelas PluginMonitoring dan file MANIFEST.MF, implementasi berikutnya adalah menghapus *.LAUNCHER* pada atribut kategori *android:name* di *androidmanifest.xml* pada modul aplikasi dan menggantinya dengan *.DEFAULT.* Hal ini bertujuan untuk menghilangkan *icon* modul aplikasi dari menu.

4.2.2 Implementasi Menampilkan Diagram Batang

Pada Tugas Akhir ini, berbagai macam data statistik yang dibutuhkan untuk *monitoring* akan ditampilkan ke dalam diagram batang. Untuk implementasi diagram batang tersebut, digunakan

library MPAndroidChart. Secara umum, implementasi menampilkan diagram batang menggunakan MPAndroidChart dapat dilihat seperti pada potongan Kode Sumber 4.3 *setBar()* berikut.

```
setBar(){
    FOR i < jumlah departemen
        yVal ← y[i]
        xVal ← x[i]
    END FOR

    BarDataSet dataSet ← yVal

    ArrayList label ← dataset

    BarData data ← (xVal, label)
    chart ← data

    tampilkan chart pada layout
}
```

Kode Sumber 4.3 *Pseudocode* Implementasi Menampilkan Diagram Batang

yVal merupakan variabel yang berfungsi untuk menyimpan nilai-nilai pada sumbu y sebuah diagram batang, sedangkan xVal adalah variabel untuk menyimpan nilai-nilai pada sumbu x. Nilai yang sisimpan dalam yVal akan disimpan ke dalam sebuah variabel BarDataSet untuk kemudian disimpan ke variabel ArrayList dan akan digabungkan dengan variabel xVal di dalam sebuah variabel BarData. Variabel BarData tersebut nantinya akan ditampilkan ke dalam diagram batang pada *layout* yang sudah disediakan.

4.2.3 Implementasi Menampilkan Diagram Lingkaran

Pada tugas akhir ini diagram lingkaran digunakan untuk menampilkan persentase jumlah tanggapan baik terhadap total jumlah tanggapan. Untuk implementasi diagram tersebut, digunakan *library* MPAndroidChart. Secara umum, implementasi menampilkan diagram menggunakan MPAndroidChart dapat

dilihat seperti pada potongan Kode Sumber 4.4 *setPieChart()* berikut.

```
setPieChart(val1, val2, val3){
    Array entries ← null
    entries ← val1
    entries ← val2
    entries ← val3

    BarDataSet dataSet ← entries

    ArrayList label ← null
    label ← "label1"
    label ← "label2"
    label ← "label3"

    PieData data ← (label, dataSet)

    chart ← data
    tampilkan chart pada layout
}
```

Kode Sumber 4.4 *Pseudocode* Implementasi Menampilkan Diagram Lingkaran

4.2.4 Implementasi Menampilkan *Listview*

Pada Tugas Akhir ini *listview* digunakan untuk menampilkan data-data statistik perdepartemen dalam bentuk daftar atau *list*. Data-data statistik yang ditampilkan meliputi jumlah aduan, jumlah aduan terjawab, jumlah aduan belum terjawab, waktu rata-rata penyelesaian aduan, rata-rata *rating*, dan jumlah tanggapan setiap departemen.

Secara garis besar, implementasi ini dilakukan dengan cara mengambil data yang ingin ditampilkan pada *listview* dari *database* melalui *web service*, kemudian data-data tersebut dimasukkan ke dalam sebuah *list* yang akan ditampilkan menggunakan kelas *adapter* yang telah dibuat. Implementasi fungsi ini dapat dilihat seperti pada potongan Kode Sumber 4.5 berikut.

```
1  inisialisasi JSONArray dengan tag "DATA"
2
3  FOR i < jumlah data JSONArray
4      JSONObject ← JSONArray[i]
5      ambil data dari JSONObject tag "ID"
6      ambil data dari JSONObject tag "NAMA_DEPT"
7      ambil data dari JSONObject tag "JML_ADUAN"
8      masukkan data ke dalam list
9  END FOR
```

Kode Sumber 4.5 *Pseudocode* Implementasi Contoh Pengambilan Data Untuk *Listview* Dalam Java

Pada Kode Sumber 4.5 diatas, data berupa nama departemen, id departemen, jumlah aduan, jumlah aduan terjawab, dan jumlah aduan belum terjawab seluruh departemen diambil dari *database* melalui *web service* untuk kemudian disimpan ke dalam sebuah *list*. Kemudian *list* tersebut akan ditampilkan menjadi daftar *listview* seperti yang ditunjukkan pada Kode Sumber 4.6 berikut.

```
1  inisialisasi adapter
2  set adapter pada variable list dengan adapter
```

Kode Sumber 4.6 *Pseudocode* Implementasi Menampilkan Listview

4.2.5 Implementasi *Search*

Fungsi *search* pada Tugas Akhir ini diimplementasikan di setiap *listview* yang menampilkan data-data departemen. Implementasi fungsi ini dapat dilihat seperti pada potongan kode program *filter()* berikut.

```
1  filter(String text){
2      ubah text ke bentuk lower case
3      clear list
4
5      IF panjang text 0 THEN
6          tampilkan semua data pada list
7      ELSE
8          FOR setiap item pada list
9              IF item.a terdapat text THEN
10                 tambahkan item ke list
11             END IF
12
```

```
13              IF item.b terdapat text THEN
14                  tambahkan item ke list
15              END IF
16          END FOR
17      END IF
18
19      refresh list
20  }
```

Kode Sumber 4.7 *Pseudocode* Implementasi *Search*

Secara garis besar, proses *search* dilakukan dengan cara membandingkan parameter text dengan setiap atribut pada *listview item* (item.id, item.a, item.b, dst.), jika ditemukan maka *listview* akan menampilkan *listview item* tersebut.

4.2.6 Implementasi *Sort*

Fungsi *sort* pada Tugas Akhir ini diimplementasikan di setiap *listview* yang menampilkan data-data departemen. Implementasi fungsi ini dapat dilihat seperti pada potongan kode sumber Java berikut.

```
1  compare(list x, list y) {
2      return x.id - y.id
3  }
4  set adapter pada variable list dengan adapter
5  refresh list
```

Kode Sumber 4.8 *Pseudocode* Implementasi *Sort* Dalam Java

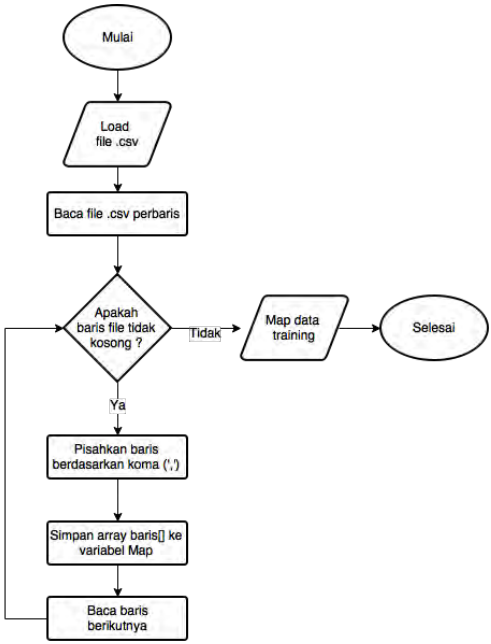
Implementasi fungsi *sort* menggunakan fungsi *native* dari *class adapter* yang telah disediakan oleh Android, dimana secara garis besar proses *sort* dilakukan dengan cara membandingkan atribut yang diinginkan pada setiap *listview item*. Fungsi akan mengembalikan nilai integer positif jika atribut item1 > atribut item2, nilai integer negatif jika atribut item1 < atribut item2, dan nilai 0 jika atribut item1 = atribut item2.

4.3 Implementasi Proses Klasifikasi

Pada Tugas Akhir ini, klasifikasi adalah proses yang berfungsi mengimplementasikan algoritma Naïve Bayes untuk mengklasifikasikan tanggapan petugas yang nantinya hasil klasifikasi tersebut akan digunakan pada kasus penggunaan 'Melihat Nilai Departemen'.

4.3.1 Implementasi *Load Data Training*

Fungsi ini bertugas untuk mengambil tanggapan-tanggapan yang digunakan sebagai data training dalam bentuk file .csv (*Comma Separated Value*). Aliran proses implementasi *load data training* digambarkan dalam Gambar 4.1 berikut.



Gambar 4.1 Diagram Alir Proses *Load Data Training*

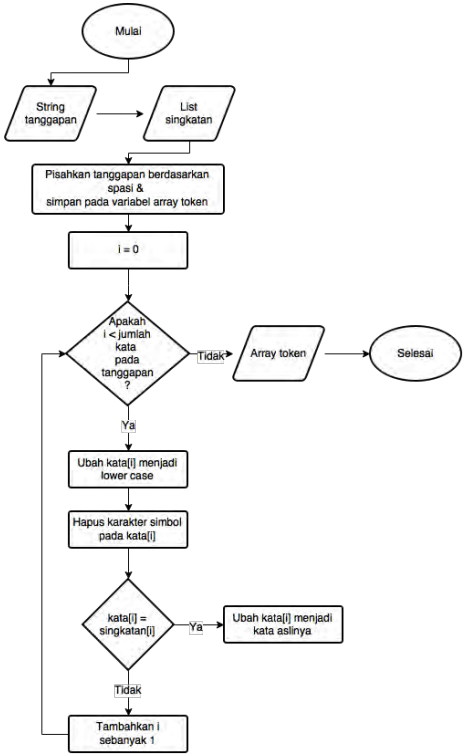
Sementara implementasi fungsi ini dapat dilihat seperti pada potongan kode program *loadDataTraining()* berikut.

```
1  loadDataTraining() {
2      file ← data_training.csv
3      readLine ← null
4      WHILE baris pada file tidak null
5         readLine ← baris pada file
6         data ← pisahkan readLine dengan ‘,’
7         Map dataTraining ← data
8      END WHILE
9
10     return dataTraining
11 }
```

Kode Sumber 4.9 *Pseudocode* Implementasi *Load Data Training*

4.3.2 Implementasi *Tokenization*

Pada tahap *tokenization* dilakukan proses pemisahan kata dari kalimat pada tanggapan, mengubah seluruh kata menjadi huruf kecil, serta menghilangkan simbol dan tanda baca. Proses pemisahan kata dilakukan dengan memisahkan setiap spasi (“ ”) dari kalimat. Selain itu, pada tahap ini juga dilakukan perubahan beberapa kalimat singkatan menjadi kalimat aslinya dengan cara membandingkan setiap kata dengan kata singkatan (item[0]) pada daftar kata singkatan (file .txt), lalu menggantinya dengan kata asli dari singkatan tersebut (item[1]). Aliran proses implementasi *tokenization* digambarkan dalam Gambar 4.2 berikut.



Gambar 4.2 Diagram Alir Proses *Tokenization*

Sementara implementasi fungsi ini dapat dilihat seperti pada potongan kode program *tokenization()* berikut.

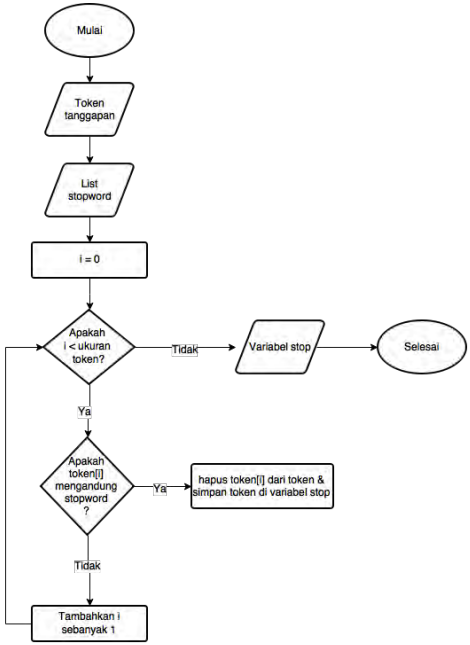
```
1  tokenization(String doc, List singkatan){
2      tokenized ← pisahkan doc dengan " "
3      i ← 0
4      WHILE i < ukuran tokenized
5         ubah tokenized[i] ke bentuk lower case
6         hapus simbol dari tokenized[i]
7
8         FOR i untuk setiap item pada array
9         singkatan
10            IF tokenized[i] = item[0] THEN
```

```
11                  tokenized[i] ← item[1]
12              END IF
13          END FOR
14          i ← i + 1
15      END WHILE
16      return tokenized
17  }
```

Kode Sumber 4.10 *Pseudocode* Implementasi *Tokenization*

4.3.3 Implementasi *Stopword Removal*

Tahap *stopword removal* dilakukan untuk menghilangkan kata-kata yang sering muncul dalam tanggapan namun tidak memiliki nilai yang berarti pada sebuah tanggapan. Proses menghapus *stopword* dilakukan dengan cara membandingkan setiap kata dengan *stopword* pada *stopword list.* Jika kata tersebut ditemukan pada *list*, maka kata tersebut akan dihilangkan dari tanggapan. *List stopword* yang digunakan dalam Tugas Akhir ini berjumlah sebanyak 357 kata [4] dan dapat dilihat pada LAMPIRAN B DAFTAR STOPWORDS. Aliran proses implementasi *stopword removal* digambarkan dalam Gambar 4.3 berikut.



Gambar 4.3 Diagram Alir Proses *Stopword Removal*

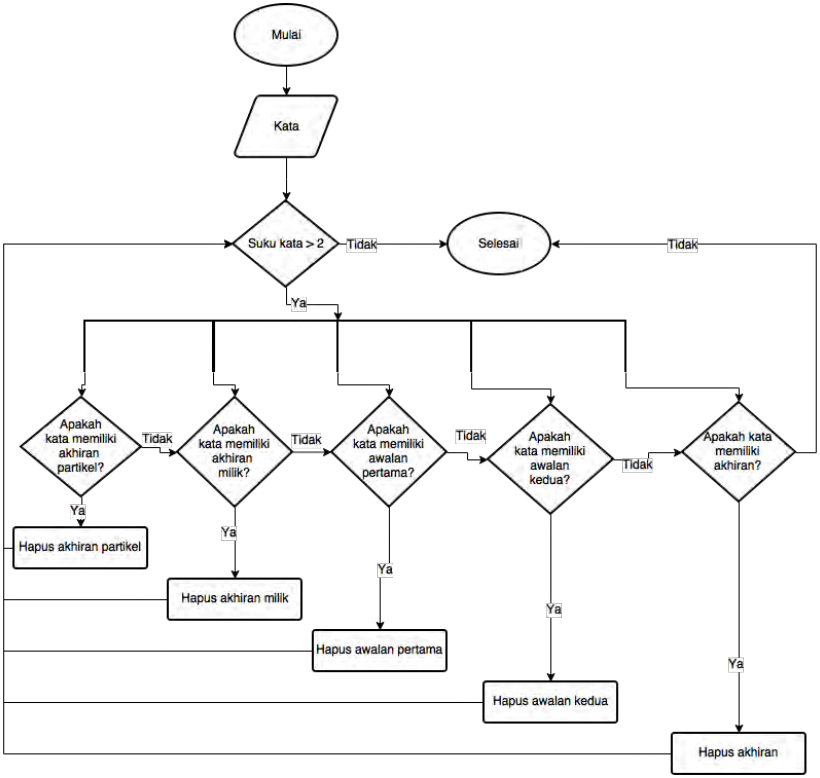
Sementara implementasi fungsi ini dapat dilihat seperti pada potongan kode program *stopwordRemoval()* berikut.

```
stopwordRemoval(String[] tokenized,
Liststopword){
    i ← 0
    WHILE i < ukuran tokenized
       IF tokenized[i] terdapat stopword THEN
           hapus tokenized[i]
       END IF
       i ← i + 1
    END WHILE
    stop ← tokenized

    return stop
}
```

Kode Sumber 4.11 *Pseudocode* Implementasi *Stopword Removal*

4.3.4 Implementasi *Stemming*

Tahap *stemming* digunakan untuk mengubah kata menjadi bentuk dasarnya. *Stemming* yang diimplementasikan pada Tugas Akhir ini menggunakan algoritma *stemming Porter* dengan cara kerja seperti yang telah dituliskan pada Bab II. Aliran proses implementasi *stemming* digambarkan dalam Gambar 4.4 berikut.



Gambar 4.4 Diagram Alir Proses *Stemming*

Sementara implementasi fungsi ini dapat dilihat seperti pada potongan kode-kode program berikut.

Potongan Kode Sumber 4.12 berikut merupakan implementasi fungsi *stem()* dengan parameter *doc* dan *length,* dimana *doc* adalah kata yang akan diubah ke bentuk dasarnya dan *length* adalah panjang kata tersebut. Algoritma Porter mencari kata dasar dengan memanfaatkan jumlah suku kata (*syllable*) pada sebuah kata seperti yang telah dijelaskan pada Bab 2.

```
 1  stem(String[] doc, int length){
 2      numSyllables ← 0
 3      kata ← doc
 4
 5      FOR i < length
 6          IF isVowel(doc[i]) bernilai true THEN
 7              numSyllables ← numSyllables + 1
 8          END IF
 9      END FOR
10
11      IF numSyllables > 2 THEN
12          length ← removeParticle(doc, length)
13      END IF
14      IF numSyllables > 2 THEN
15          length ← removePossessivePronoun(doc,
16          length)
17      END IF
18      IF numSyllables > 2 THEN
19          length ← removeFirstOrderPrefix(doc,
20          length)
21      END IF
22      IF numSyllables > 2 THEN
23          length ← removeSecondOrderPrefix(doc,
24          length)
25      END IF
26      IF numSyllables > 2 THEN
27          length ← removeSuffix(doc, length)
28      END IF
29
30      kataDasar ← kata
31      return kataDasar
32  }
```

Kode Sumber 4.12 *Pseudocode* Implementasi fungsi *stem()*

Potongan Kode Sumber 4.13 berikut adalah implementasi fungsi *isVowel()* yang berfungsi untuk mengecek keberadaan huruf vokal pada sebuah kata. Fungsi ini akan digunakan pada fungsi *stem*() untuk menghitung jumlah suku kata dari suatu kata.

```
1  isVowel(char c){
2      IF c = 'a' OR c = 'i' OR c = 'u' OR c =
3      'e' OR c = 'o' THEN
4          return true
5
6      ELSE
7          return false
8      END IF
9  }
```

Kode Sumber 4.13 *Pseudocode* Implementasi fungsi *isVowel()*

Potongan Kode Sumber 4.14 berikut adalah implementasi fungsi *removePossessivePronoun()* yang berfungsi untuk menghilangkan kata akhiran milik pada sebuah kata.

```
1  removePossessivePronoun(String t, int length){
2      IF t berakhiran 'ku' OR t berakhiran 'mu'
3      THEN
4          numSyllables ← numSyllables – 1
5          kata ← hapus 2 huruf terakhir pada t
6          return length - 2
7      END IF
8
9      IF t berakhiran 'nya' THEN
10         numSyllables ← numSyllables – 1
11         kata ← hapus 3 huruf terakhir pada t
12         return length - 3
13     END IF
14
15     return length
16 }
```

Kode Sumber 4.14 *Pseudocode* Implementasi fungsi *removePossessivePronoun()*

Potongan Kode Sumber 4.15 berikut adalah implementasi fungsi *removeParticle()* yang berfungsi untuk menghilangkan kata akhiran partikel pada sebuah kata.

```
removeParticle(String t, int length){
    IF t berakhiran 'kah' OR t berakhiran
    'lah' OR t berakhiran 'pun' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 3 huruf terakhir pada t
        return length - 2
    END IF
    return length
}
```

Kode Sumber 4.15 *Pseudocode* Implementasi fungsi *removeParticle()*

Potongan Kode Sumber 4.16 berikut adalah implementasi fungsi *removeFirstOrderPreffix()* yang berfungsi untuk menghilangkan kata awalan pertama pada sebuah kata.

```
removeFirstOrderPreffix(String t, int length){
    IF t berawalan 'meng' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 4 kata pertama pada t
        return length - 4
    END IF
    IF t berawalan 'meny' AND length > 4 AND
    isVowel(t[4]) bernilai true THEN
        t[3] ← 's'
        numSyllables ← numSyllables – 1
        kata ← hapus 3 kata pertama pada t
        return length - 3
    END IF
    IF t berawalan 'men' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 3 kata pertama pada t
        return length - 3
    END IF
    . . .
    return length
}
```

Kode Sumber 4.16 *Pseudocode* Implementasi fungsi *removeFirstOrderPreffix()*

Potongan Kode Sumber 4.17 berikut adalah implementasi fungsi *removeSecondOrderPreffix()* yang berfungsi untuk menghilangkan kata awalan kedua pada sebuah kata.

```
removeSecondOrderPreffix(String t, int
length){
    IF t berawalan 'ber' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 3 kata pertama pada t
        return length - 3
    END IF

    IF t berawalan 'be' AND length > 4 AND
    isVowel(t[2]) bernilai false AND t[3] =
    'e' AND t[4] = 'r' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 2 kata pertama pada t
        return length - 2
    END IF

    IF t berawalan 'per' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 3 kata pertama pada t
        return length - 3
    END IF

    IF t berawalan 'pe' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 2 kata pertama pada t
        return length - 2
    END IF

    return length
}
```

Kode Sumber 4.17 *Pseudocode* Implementasi fungsi *removeSecondOrderPreffix()*

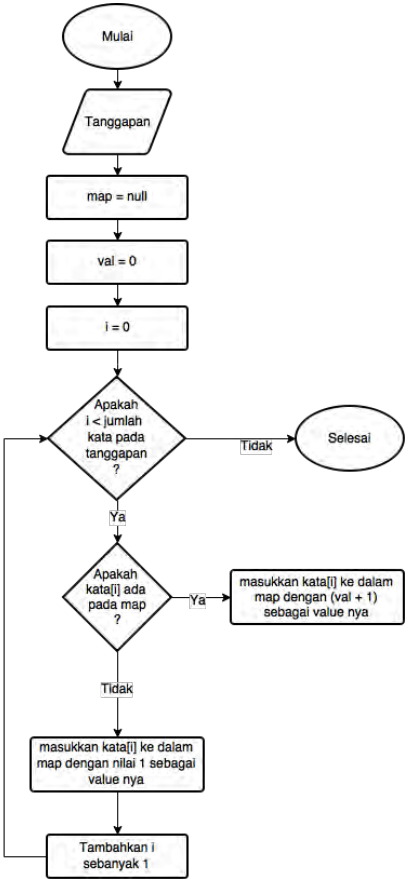
Potongan Kode Sumber 4.18 berikut adalah implementasi fungsi *removeSuffix()* yang berfungsi untuk menghilangkan kata akhiran pada sebuah kata.

```
removeSuffix(String t, int length){
    IF t berakhiran 'kan' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 3 kata terakhir pada t
        return length - 3
    END IF

    IF t berakhiran 'an' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus 2 kata terakhir pada t
        return length - 2
    END IF

    IF t berakhiran 'i' AND  t not ends with
    'si' THEN
        numSyllables ← numSyllables – 1
        kata ← hapus kata terakhir pada t
        return length - 1
    END IF

    return length
}
```

Kode Sumber 4.18 *Pseudocode* Implementasi fungsi *removeSuffix()*

4.3.5 Implementasi *Indexing*

Proses *indexing* berfungsi untuk menghitung nilai kemunculan kata dalam tiap tanggapan. Penentuan nilai tanggapan dapat dilakukan dengan menghitung frekuensi suatu kata pada tanggapan. Frekuensi berfungsi untuk menentukan seberapa sering suatu kata muncul dalam sebuah tanggapan. Nilai frekuensi ini kemudian akan digunakan untuk proses menghitung probabilitas pada algoritma Naïve Bayes. Aliran proses implementasi *indexing* digambarkan dalam Gambar 4.5 berikut.



Gambar 4.5 Diagram Alir Proses *Indexing*

Sementara implementasi fungsi *indexing* ini dapat dilihat seperti pada potongan Kode Sumber 4.19 *termFrequency()* berikut.

```
termFrequency(List doc, String kategori){
    Map map ← null
    Map temp ← null
    val ← 0
    FOR i < ukuran doc
        IF map terdapat doc[i] THEN
            map ← (doc[i], val + 1)
        ELSE
            map ← (doc[i], 1)
        END IF
    END FOR

    FOR i < ukuran map THEN
        data ← map[i] + kategori
        temp ← data
    END FOR

    return temp
}
```

Kode Sumber 4.19 *Pseudocode* Implementasi *indexing*

4.3.6 Implementasi Klasifikasi

Implementasi klasifikasi dalam Tugas Akhir ini menggunakan algoritma Naïve Bayes. Konsepnya yaitu dengan menghitung nilai probabilitas semua tanggapan dari tiap-tiap kategori yang ada. Setelah didapat semua nilai probabilitas tersebut, berikutnya dilakukan perkalian untuk setiap dokumen sejumlah kategori yang ada. Dari hasil perkalian probabilitas tersebut, diambil nilai terbsesar untuk menentukan sebuah tanggapan termasuk kategori yang mana.

Aliran proses implementasi klasifikasi digambarkan dalam Gambar 4.6 berikut.

Gambar 4.6 Diagram Alir Proses Klasifikasi

Sementara implementasi klasifikasi ini terlihat pada potongan Kode Sumber 4.20 fungsi *naiveBayes()* berikut. Dimana *calculatePriors()* adalah fungsi untuk menghitung nilai probabilitas kemunculan suatu kategori seperti pada Persamaan (2.3) yang telah dijelaskan pada Bab II, sedangkan *calculateConditionalProbabilities()* adalah fungsi untuk

menghitung probabilitas kemunculan suatu kata pada sebuah tanggapan terhadap suatu kategori seperti pada Persamaan (2.4) yang telah dijelaskan pada Bab II. Sementara *chooseCategory()* adalah fungsi yang akan melakukan perkalian terhadap nilai-nilai probabilitas yang sudah dihitung dan kemudian akan mengategorikan sebuah tanggapan termasuk ke kategori mana sesuai dengan nilai yang terbesar, seperti pada Persamaan (2.2) yang telah dijelaskan pada Bab II.

```
 1  naiveBayes(String tanggapan, Map dataTraining){
 2      //praproses data training
 3      FOR i < ukuran dataTraining
 4          token ← tokenization(dataTraining[0],
 5          singkatan)
 6          stopword ← stopwordRemoval(token,
 7          stopwords)
 8          FOR i < ukuran stopword
 9              stemmedDoc ← stem(stopword[i],
10              stopword.length)
11          END FOR
12          tf ← termFrequency(stemmedDoc,
13          dataTraining[1])
14      END FOR
15
16      //praproses data testing
17      token ← tokenization(tanggapan, singkatan)
18      stopword ← stopwordRemoval(token,
20      stopwords)
21      FOR i < ukuran stopword
22          stemmedDoc ← stem(stopword[i],
23          stopword.length)
24      END FOR
25      dTest ← termFrequency(stemmedDoc, null)
26
27      FOR i < ukuran category
28          calculateConditionalProbabilities
29          (dTest, category1)
30
31          calculateConditionalProbabilities
32          (dTest, category2)
33
34          calculateConditionalProbabilities
```

```
35          (dTest, category3)
36      END FOR
37
38      hasil ← chooseCategory(dTest,
39      calculatePriors(dataTraining))
40
41      return hasil
42  }
```

Kode Sumber 4.20 *Pseudocode* Implementasi klasifikasi

4.4 Implementasi *Query* Basis Data

Pada subbab ini akan dijelaskan mengenai implementasi *query* yang digunakan untuk mengakses *server*, *query* ini disimpan dalam suatu *web service* berbahasa PHP pada *server*. *Query* yang diimplementasikan dibuat menggunakan bahasa SQL.

4.4.1 Implementasi Query Menampilkan Aduan

Query menampilkan aduan adalah *query* yang berfungsi untuk menampilkan data jumlah total aduan yang diterima, jumlah aduan yang sudah dijawab, dan jumlah aduan yang belum dijawab. Implementasi *query-query* tersebut ditunjukkan pada Kode Sumber 4.21, Kode Sumber 4.22, Kode Sumber 4.23, dan Kode Sumber 4.24.

Query untuk menampilkan jumlah total aduan yang masuk dilakukan dengan cara menghitung jumlah baris pada tabel aduan ketika aduan tidak termasuk *spam*. Implementasi query tersebut dapat dilihat pada Kode Sumber 4.21 berikut.

```
1  SELECT COUNT(*) AS masuk
2  FROM aduan
3  WHERE aduan.spam = 0
```

Kode Sumber 4.21 Implementasi *Query* Menampilkan Jumlah Total Aduan Masuk

Query untuk menampilkan jumlah total aduan terjawab dilakukan dengan cara menghitung jumlah baris pada tabel aduan ketika atribut *status* bernilai 4, aduan bukan *spam*, dan atribut *info*

tidak bernilai 2. Implementasi *query* tersebut dapat dilihat pada Kode Sumber 4.22 berikut.

```
1  SELECT COUNT(*) AS terjawab
2  FROM aduan
3  WHERE status = 4
4      AND aduan.spam = 0
5      AND info <> (2)
```

Kode Sumber 4.22 Implementasi *Query* Menampilkan Jumlah Total Aduan Terjawab

Query untuk menampilkan jumlah total aduan belum terjawab dilakukan dengan cara menghitung jumlah baris pada tabel aduan ketika atribut *status* tidak bernilai 4, aduan bukan *spam*, dan atribut *info* tidak bernilai 2. Implementasi *query* tersebut dapat dilihat pada Kode Sumber 4.23 berikut.

```
1  SELECT COUNT(*) AS belum_terjawab
2  FROM aduan
3  WHERE status <> (4)
4      AND aduan.spam = 0
5      AND info <> (2)
```

Kode Sumber 4.23 Implementasi *Query* Menampilkan Jumlah Total Aduan Belum Terjawab

Query untuk menampilkan data aduan untuk setiap departemen secara garis besar dilakukan dengan cara yang sama dengan query pada Kode Sumber 4.21, Kode Sumber 4.22, dan Kode Sumber 4.23, namun perlu ditambahkan parameter berupa id departemen yang ada. Implementasi *query* tersebut dapat dilihat pada Kode Sumber 4.24 berikut.

```
1  SELECT terjawab.id,
2      departemen.nama_departemen AS departemen,
3      COUNT(aduan.departemen) AS jml_masuk,
4      terjawab.jml_terjawab,
5      terjawab.jml_belum_terjawab
6  FROM
7  (
8      (
9          SELECT departemen.id_departemen AS id,
```

```
10              COUNT(aduan.departemen) AS
11              jml_terjawab,
12              belum.jml_belum_terjawab AS
13              jml_belum_terjawab
14          FROM
15          (
16              SELECT departemen.id_departemen
17                  AS id,
18                  COUNT(aduan.departemen) AS
19                  jml_belum_terjawab
20              FROM departemen
21              LEFT JOIN aduan
22              ON aduan.departemen =
23                  departemen.id_departemen
24                  AND aduan.status NOT IN (4)
25                  AND aduan.spam = 0
26                  AND aduan.info <> (2)
27              GROUP BY departemen.nama_departemen
28          ) belum
29          INNER JOIN departemen
30          ON belum.id = departemen.id_departemen
31          LEFT JOIN aduan
32          ON aduan.departemen =
33              departemen.id_departemen
34              AND aduan.status = 4
35              AND aduan.spam = 0
36              AND aduan.info <> (2)
37          GROUP BY departemen.nama_departemen
38      ) terjawab
39      INNER JOIN departemen
40      ON terjawab.id = departemen.id_departemen
41  )
42  LEFT JOIN aduan
43  ON aduan.departemen = departemen.id_departemen
44  AND aduan.departemen = terjawab.id
45  AND aduan.spam = 0
46  GROUP BY departemen.nama_departemen
47  ORDER BY terjawab.id;
```

Kode Sumber 4.24 Implementasi *Query* Menampilkan Data Aduan Setiap Departemen

Selain itu terdapat *query* untuk menampilkan data aduan perbulan, dimana pengguna dapat melihat data statistik aduan yang

diterima suatu departemen pada bulan tertentu. Pada *query* ini, aplikasi akan mengirimkan data bulan dan tahun yang dipilih pengguna pada halaman Statistik Aduan, kemudian *web service* akan mengeksekusi *query* tersebut untuk mengambil data dari *database*. Implementasi *query* untuk menampilkan data aduan perbulan ditunjukkan pada Kode Sumber 4.25.

```
SELECT departemen.id_departemen,
    departemen.nama_departemen AS departemen,
    COUNT(aduan.departemen) AS jml_masuk,
    terjawab.jml_terjawab,
    terjawab.jml_belum_terjawab
FROM
    (
        (
            SELECT departemen.id_departemen
                AS id, COUNT(aduan.departemen)
                AS jml_terjawab,
                belum.jml_belum_terjawab
                AS jml_belum_terjawab
            FROM (
                SELECT
                    departemen.id_departemen
                    AS id,
                    COUNT(aduan.departemen) AS
                    jml_belum_terjawab
                FROM departemen
                LEFT JOIN aduan
                ON aduan.departemen =
                    departemen.id_departemen
                    AND aduan.status NOT IN(4)
                    AND aduan.spam = 0
                    AND aduan.info <> (2)
                    AND MONTH(aduan.waktu) =
                    '$month'
                    AND YEAR(aduan.waktu) =
                    '$year'
                GROUP BY
                    departemen.nama_departemen,
                    MONTH(aduan.waktu)
        ) belum
        INNER JOIN departemen
        ON belum.id = departemen.id_departemen
```

```
37          LEFT JOIN aduan
38          ON aduan.departemen =
39              departemen.id_departemen AND
40              aduan.status = 4 AND aduan.spam = 0
41              AND aduan.info <> (2) AND
42              MONTH(aduan.waktu) = '$month' AND
43              YEAR(aduan.waktu) = '$year'
44          GROUP BY departemen.nama_departemen,
45              MONTH(aduan.waktu)
46      ) terjawab
47      INNER JOIN departemen
48      ON terjawab.id = departemen.id_departemen
49  )
50  LEFT JOIN aduan
51  ON aduan.departemen = departemen.id_departemen
52      AND aduan.departemen = terjawab.id
53      AND aduan.spam = 0
54      AND MONTH(aduan.waktu) = '$month'
55      AND YEAR(aduan.waktu) = '$year'
56  GROUP BY departemen.nama_departemen,
57      MONTH(aduan.waktu)
58  ORDER BY departemen.id_departemen;
```

Kode Sumber 4.25 Implementasi *Query* Menampilkan Data Aduan Perbulan

4.4.2 Implementasi Query Menampilkan Aduan Belum Terjawab

Query menampilkan aduan belum terjawab adalah *query* yang berfungsi untuk menampilkan data jumlah aduan yang belum dijawab oleh suatu departemen. Pada *query* ini, aplikasi akan mengirimkan data berupa id departemen yang dipilih pengguna dari halaman Statistik Aduan, kemudian *web service* akan mengeksekusi *query* dengan cara mengambil data aduan dengan status tidak bernilai 4, bukan spam, info tidak bernilai 2. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.26.

```
1   SELECT aduan.id_aduan AS id,
2       aduan.topik, aduan.isi, aduan.waktu,
3       status.nama_status AS status,
4       prioritas.nama_prioritas AS prioritas
5   FROM aduan, status, departemen, prioritas
```

```
WHERE aduan.status = status.id_status AND
    aduan.departemen=departemen.id_departemen
    AND departemen.id_departemen='$departemen'
    AND aduan.status <> (4)
    AND aduan.spam = 0
    AND info <> (2)
    AND aduan.prioritas=prioritas.id_prioritas
ORDER BY aduan.id_aduan;
```

Kode Sumber 4.26 Implementasi *Query* Menampilkan Aduan Belum Terjawab

4.4.3 Implementasi Query Menampilkan Rerata Waktu

Query menampilkan rerata waktu adalah *query* yang berfungsi untuk menampilkan data rata-rata waktu yang dibutuhkan suatu departemen untuk menanggapi aduan yang diterima. *Query* ini diimplementasikan dengan cara mengambil rata-rata pada atribut waktu di tabel aduan dengan *group by* setiap departemen. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.27.

Query ini akan dieksekusi secara otomatis oleh *web service* untuk mengambil data dari *database* ketika pengguna mengakses halaman Statistik Waktu pada aplikasi.

```
SELECT departemen.id_departemen,
    departemen.nama_departemen AS departemen,
    x.waktu
FROM
(
    SELECT id_departemen as id,
        round(avg(TIMESTAMPDIFF(HOUR,
        aduan.waktu,
        detail_aduan.waktu_detail)), 2)
        as waktu
    FROM aduan
    LEFT JOIN departemen
    ON aduan.departemen =
        departemen.id_departemen
    LEFT JOIN detail_aduan
    ON detail_aduan.aduan = aduan.id_aduan
    WHERE aduan.status = 4
        AND spam = 0
```

```
19          AND detail_aduan.waktu_detail =
20          (
21              SELECT waktu_detail
22              FROM detail_aduan
23              WHERE detail_aduan.aduan =
24                  aduan.id_aduan
25              ORDER BY waktu_detail DESC
26              LIMIT 1
27          )
28          GROUP BY id_departemen
29  ) x
30  RIGHT JOIN departemen
31  ON x.id = departemen.id_departemen
32  GROUP BY departemen.nama_departemen
33  ORDER BY departemen.id_departemen;
```

Kode Sumber 4.27 Implementasi *Query* Menampilkan Rerata Waktu

4.4.4 Implementasi Query Menampilkan Rerata Rating

Query menampilkan rerata rating adalah *query* yang berfungsi untuk menampilkan data rata-rata rating yang dimiliki suatu departemen. *Query* ini diimplementasikan dengan cara mengambil rata-rata pada atribut rating di tabel aduan dengan *group by* setiap departemen. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.28.

Query ini akan dieksekusi secara otomatis oleh *web service* untuk mengambil data dari *database* ketika pengguna mengakses halaman Statistik Rating pada aplikasi.

```
1   SELECT departemen.id_departemen,
2       departemen.nama_departemen as departemen,
3       round(AVG(aduan.rating), 2) as rating
4   FROM aduan
5   RIGHT JOIN departemen
6   ON aduan.departemen = departemen.id_departemen
7   GROUP BY departemen.nama_departemen
8   ORDER BY departemen.id_departemen;
```

Kode Sumber 4.28 Implementasi *Query* Menampilkan Rerata Rating

4.4.5 Implementasi Query Menampilkan Tanggapan Departemen

Query menampilkan tanggapan departemen adalah *query* yang berfungsi untuk menampilkan seluruh tanggapan yang diberikan oleh suatu departemen. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.29 dan Kode Sumber 4.30.

Query pada Kode Sumber 4.29 akan dieksekusi secara otomatis oleh *web service* untuk mengambil data dari *database* ketika pengguna mengakses halaman Tanggapan Departemen pada aplikasi. Data keluaran dari *query* tersebut adalah seluruh departemen beserta jumlah tanggapan yang dimiliki departemen tersebut, yang kemudian akan ditampilkan ke dalam *listview* pada halaman Tanggapan Departemen. *Query* ini diimplementasikan dengan cara mengambil jumlah tanggapan setiap departemen pada detail_aduan. Pada tabel detail_aduan, tanggapan ditunjukkan dengan atribut petugas_detail yang tidak *null.*

```
 1  SELECT departemen.id_departemen,
 2      departemen.nama_departemen,
 3      y.jml_tanggapan
 4  FROM departemen
 5  LEFT JOIN
 6  (
 7      SELECT departemen.id_departemen,
 8          sum(x.jumlah) AS jml_tanggapan
 9      FROM petugas, departemen,
10      (
11          SELECT count(*) AS jumlah,
12              id_detail_aduan,
13              petugas_detail
14          FROM detail_aduan
15          WHERE petugas_detail IS NOT NULL
16              AND YEAR(waktu_detail) = 2015
17          GROUP BY petugas_detail
18      ) x
19      WHERE x.petugas_detail =
20          petugas.id_petugas
21          AND petugas.departemen =
22          departemen.id_departemen
23      GROUP BY petugas.departemen
```

```
24 ) y
25 ON departemen.id_departemen = y.id_departemen;
```

Kode Sumber 4.29 Implementasi *Query* Jumlah Tanggapan Seluruh Departemen

Sedangkan pada *query* di Kode Sumber 4.30, aplikasi akan mengirimkan data berupa id departemen yang dipilih pengguna dengan cara memilih departemen pada *listview* di halaman Tanggapan Departemen, kemudian *web service* akan mengeksekusi *query* tersebut untuk mengambil data dari *database.* Data keluaran dari *query* tersebut adalah tanggapan-tanggapan yang dimiliki departemen terpilih, yang kemudian akan ditampilkan ke dalam *listview*. *Query* ini diimplementasikan dengan cara mengambil tanggapan pada detail_aduan milik departemen tertentu sesuai parameter yang diterima.

```
 1 SELECT *
 2 FROM detail_aduan, petugas
 3 WHERE YEAR(detail_aduan.waktu_detail) = 2015
 4     AND detail_aduan.petugas_detail IN
 5     (
 6         SELECT id_petugas
 7         FROM petugas
 8         WHERE departemen = '$id_dept'
 9     )
10     AND detail_aduan.petugas_detail =
11     petugas.id_petugas;
```

Kode Sumber 4.30 Implementasi *Query* Menampilkan Tanggapan Departemen

4.4.6 Implementasi Query Menyimpan Hasil Klasifikasi

Query menyimpan hasil klasifikasi adalah *query* yang berfungsi untuk menyimpan data hasil klasifikasi dari suatu tanggapan ke dalam *database*. Aplikasi akan mengirimkan data-data tanggapan yang perlu disimpan setelah proses klasifikasi tanggapan selesai, kemudian *web service* akan mengeksekusi *query* ini untuk menyimpan data ke dalam *database*. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.31.

```
INSERT INTO tanggapan
VALUES
(
    '$id_tanggapan', '$isi_tanggapan',
    '$waktu_tanggapan', '$id_petugas',
    '$nama_petugas', '$dept_petugas',
    '$nbayes', '$poin_tanggapan'
);
```

Kode Sumber 4.31 Implementasi *Query* Menyimpan Hasil Klasifikasi

4.4.7 Implementasi Query Menampilkan *Score* Departemen

Query menampilkan *Score* Departemen adalah *query* yang berfungsi untuk menampilkan nilai yang dimiliki suatu departemen berdasarkan hasil klasifikasi tanggapan-tanggapan yang dimilikinya. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.32 dan Kode Sumber 4.33.

Pada Kode Sumber 4.32, keluaran yang didapatkan adalah *score* seluruh departemen yang akan ditampilkan dalam diagram batang. *Query* ini akan dieksekusi secara otomatis oleh *web service* untuk mengambil data dari *database* ketika pengguna mengakses halaman *Scoring* pada aplikasi. *Query* ini diimplementasikan dengan cara mengambil seluruh *score* dari setiap departemen yang ada. Total *score* dihitung menggunakan fungsi *sum* pada atribut poin_tanggapan untuk setiap departemen.

```
SELECT d.id_departemen, d.nama_departemen,
   COALESCE(SUM(t.poin_tanggapan), 0) as score
FROM tanggapan t
RIGHT JOIN departemen d
ON t.id_dept_petugas =
    d.id_departemen
GROUP BY d.id_departemen
ORDER BY score DESC;
```

Kode Sumber 4.32 Implementasi *Query* Menampilkan *Score* Seluruh Departemen

Sedangkan pada Kode Sumber 4.33, keluaran yang didapatkan adalah data tanggapan hasil klasifikasi dari suatu

departemen tertentu. Pada *query* ini, aplikasi akan mengirimkan data berupa id departemen yang dipilih pengguna, kemudian *web service* akan mengeksekusi *query* tersebut untuk mengambil data dari *database*. *Query* ini diimplementasikan dengan cara mengambil jumlah tanggapan berkategori baik, netral, dan tidak baik pada kolomkategori_tanggapan untuk setiap departemen.

```
SELECT id_dept_petugas id, COUNT(*) as
tanggapan, (
    SELECT COUNT(*)
    FROM tanggapan t
    WHERE t.kategori_tanggapan = 'BAIK'
        AND t.id_dept_petugas = id
) as baik, (
    SELECT COUNT(*)
    FROM tanggapan t
    WHERE t.kategori_tanggapan = 'TIDAK BAIK'
        AND t.id_dept_petugas = id
) as tidak_baik, (
    SELECT COUNT(*)
    FROM tanggapan t
    WHERE t.kategori_tanggapan = 'NETRAL'
        AND t.id_dept_petugas = id
) as netral
FROM tanggapan
WHERE id_dept_petugas = '$id_dept'
GROUP BY id_dept_petugas;
```

Kode Sumber 4.33 Implementasi *Query* Menampilkan Statistik Tanggapan Perdepartemen

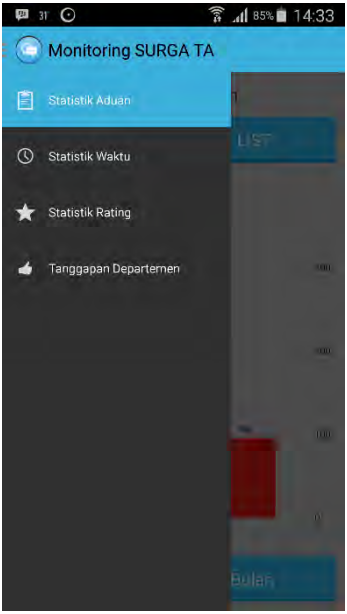
4.5 Implementasi Antarmuka

Pada subbab ini dijelaskan implementasi tampilan antarmuka yang telah dibahas pada subbab 3.2.6.

4.5.1 Implementasi Antarmuka Navigasi

Antarmuka navigasi merupakan antarmuka yang berfungsi sebagai kontrol untuk pengguna dalam mengakses menu-menu yang ada pada modul aplikasi ini, yaitu menu statistik aduan, statistik waktu, statistik rating, dan tanggapan departemen.

Antarmuka navigasi diakses dengan cara menarik jari dari sisi kiri layar perangkat bergerak pengguna.. Antarmuka ini diimplementasikan dengan kode XML dan Java sehingga membentuk tampilan sesuai Gambar 4.7.



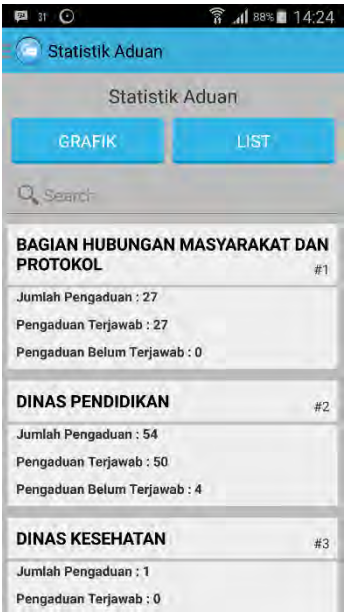
Gambar 4.7 Antarmuka Navigasi

4.5.2 Implementasi Antarmuka Statistik Aduan

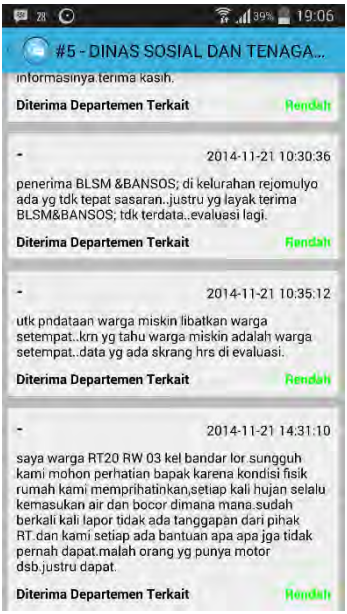
Antarmuka statistik aduan merupakan antarmuka pertama yang muncul ketika pengguna mengakses modul aplikasi ini. Pada antarmuka ini pengguna dapat melihat statistik aduan secara keseluruhan seperti pada Gambar 4.8, melihat statistik aduan perbulan seperti pada Gambar 4.9, melihat statistik aduan dalam bentuk list seperti pada Gambar 4.10, serta melihat tanggapan mana saja yang belum dijawab oleh suatu departemen seperti pada Gambar 4.11. Antarmuka menampilkan lokasi diimplementasikan menggunakan kode XML dan Java.

Gambar 4.8 Antarmuka Diagram Statistik Aduan

Gambar 4.9 Antarmuka Diagram Statistik Aduan Perbulan



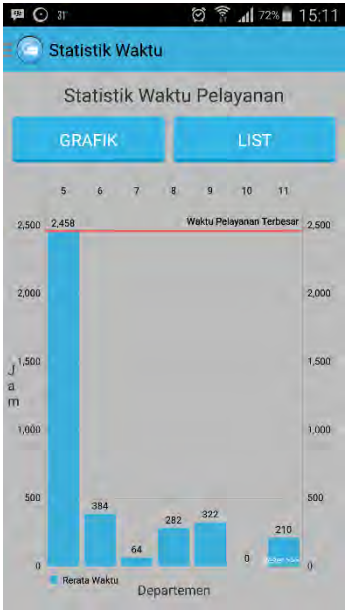
Gambar 4.10 Antarmuka List Statistik Aduan



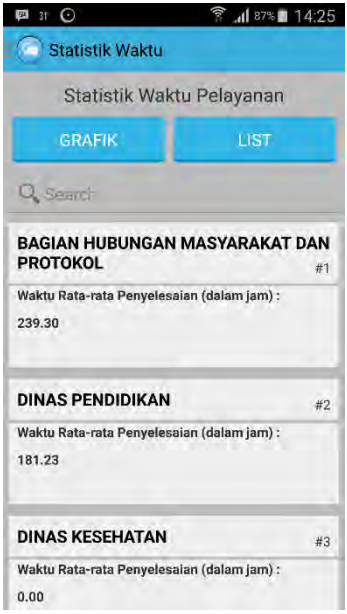
Gambar 4.11 Antarmuka Tanggapan Belum Terjawab

4.5.3 Implementasi Antarmuka Statistik Waktu

Antarmuka statistik waktu merupakan antarmuka yang memungkinkan pengguna melihat statistik rata-rata waktu pelayanan suatu departemen melakukan tanggapan dalam bentuk diagram batang seperti pada Gambar 4.12 dan dalam bentuk list seperti pada Gambar 4.13. Antarmuka ini diimplementasikan menggunakan kode XML dan Java.



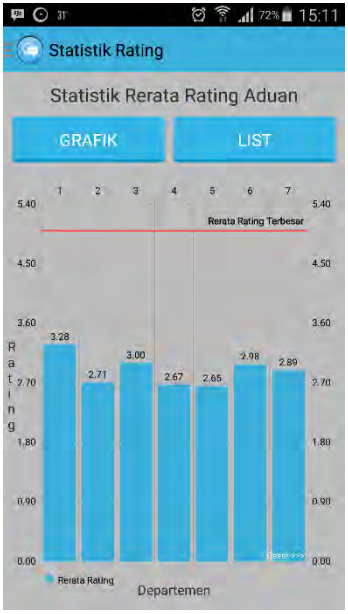
Gambar 4.12 Antarmuka Diagram Statistik Waktu



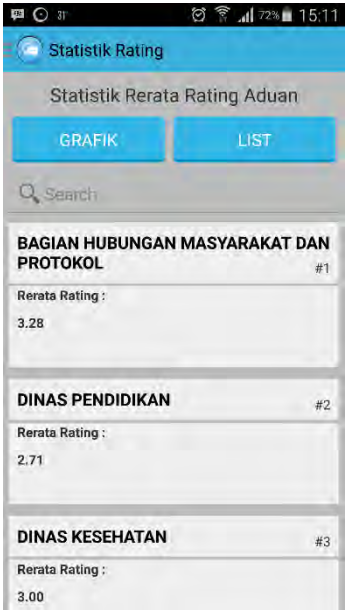
Gambar 4.13 Antarmuka List Statistik Waktu

4.5.4 Implementasi Antarmuka Statistik Rating

Antarmuka statistik rating merupakan antarmuka yang memungkinkan pengguna melihat statistik rata-rata rating suatu departemen dalam bentuk diagram batang seperti pada Gambar 4.15 dan dalam bentuk list seperti pada Gambar 4.14. Antarmuka ini diimplementasikan menggunakan kode XML dan Java.



Gambar 4.15 Antarmuka Diagram Statistik Rating



Gambar 4.14 Antarmuka List Statistik Rating

4.5.5 Implementasi Antarmuka Tanggapan Departemen

Antarmuka tanggapan departemen adalah antarmuka pertama dari proses klasifikasi yang memungkinkan pengguna untuk melakukan klasifikasi terhadap tanggapan yang dimiliki suatu departemen. Ketika mengakses antarmuka ini akan ditampilkan list departemen dan jumlah tanggapan yang dimilikinya seperti pada Gambar 4.17. Kemudian dengan memilih departemen yang diinginkan, pengguna dapat melihat tanggapan departemen tersebut Gambar 4.16 dan melakukan klasifikasi baik secara satu persatu ataupun secara keseluruhan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java.

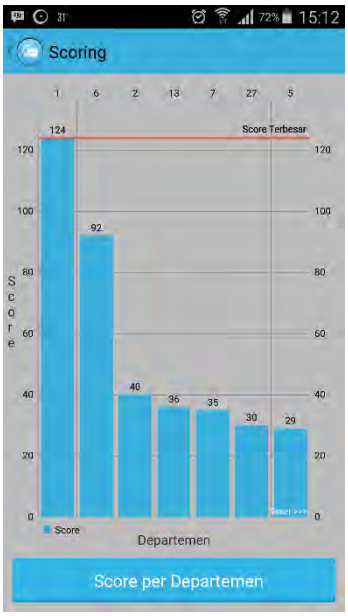
Melalui antarmuka ini pula pengguna dapat mengakses menu untuk melihat *score* departemen dengan menekan tombol menu di perangkat Android pengguna.

Gambar 4.17 Antarmuka *List* Jumlah Tanggapan Departemen

Gambar 4.16 Antarmuka *List* Tanggapan Departemen

4.5.6 Implementasi Antarmuka *Scoring*

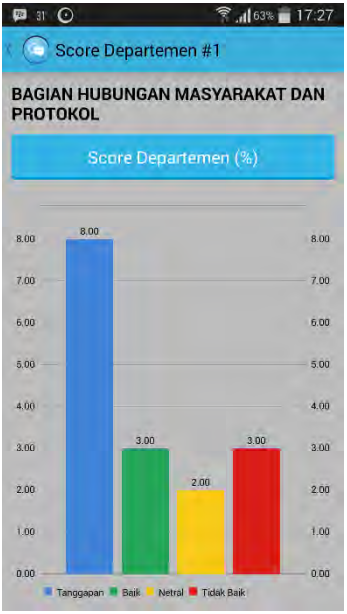
Antarmuka *scoring* atau melihat nilai adalah antarmuka yang berfungsi untuk menampilkan nilai yang dimiliki departemen dari hasil proses klasifikasi terhadap tanggapan yang dimilikinya, baik secara keseluruhan seperti pada Gambar 4.18 maupun nilai perdepartemen dengan menekan tombol yang ada di antarmuka pada Gambar 4.18. Setelah itu, untuk melihat nilai perdepartemen, pengguna dapat memilih departemen dari *list* seperti pada Gambar 4.19



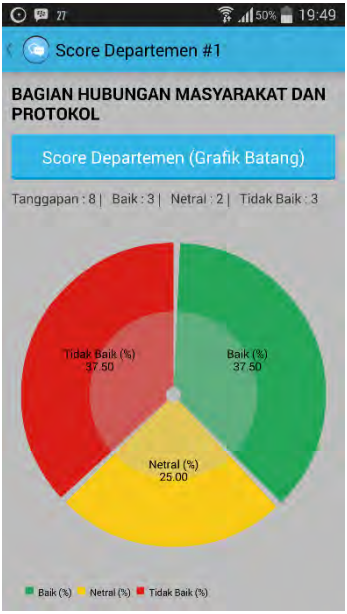
Gambar 4.18 Antarmuka Diagram *Score*

Gambar 4.19 Antarmuka List Departemen

Antar muka pada Gambar 4.21 menunjukkan nilai perdepartemen dalam bentuk diagram batang. Pengguna dapat merubah diagram yang ditampilkan menjadi diagram lingkaran seperti pada Gambar 4.20 dengan menekan tombol pada Gambar 4.21. Antarmuka ini diimplementasikan menggunakan kode XML dan Java.



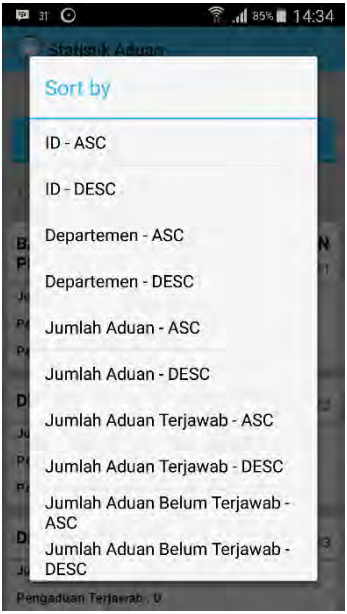
Gambar 4.21 Antarmuka Diagram Batang *Score* Departemen



Gambar 4.20 Antarmuka Diagram Lingkaran *Score* Departemen

4.5.7 Implementasi Antarmuka *Sort*

Antarmuka *sort* merupakan antar muka untuk mengurutkan atribut pada *list* departemen sesuai dengan keinginan pengguna. Antarmuka ini dapat diakses dengan menekan tombol menu pada perangkat Android pengguna di setiap antarmuka yang menampilkan *list* departemen. Antarmuka ini diimplementasikan menggunakan kode XML dan Java sehingga menjadi seperti Gambar 4.22.



Gambar 4.22 Antarmuka *Sort* Departemen

BAB V
UJI COBA DAN EVALUASI

Pada bab ini dijelaskan mengenai rangkaian uji coba dan evaluasi yang dilakukan terhadap sistem yang dibuat. Pembahasan yang dipaparkan meliputi lingkungan uji coba, data uji coba, skenario uji coba, hasil uji coba, dan evaluasi.

5.1 Lingkungan Uji Coba

Proses uji coba terhadap modul aplikasi ini dilakukan menggunakan *smartphone* dengan spesifikasi seperti berikut.

- Perangkat Bergerak Samsung Galaxy Note 3 GT-N900
  - Sistem Operasi: Android v5.0 (Lollipop),
  - CPU: 1.9 GHz Exynos
  - Memory internal: 32 GB,
  - RAM: 3.00 GB,
  - Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps, dan
  - WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot
- Perangkat Bergerak Samsung Galaxy Core Duos I8262
  - Sistem Operasi: Android v4.1 (Jelly Bean),
  - CPU: 1.2 GHz Cortex-A5
  - Memory internal: 8 GB,
  - RAM: 1.00 GB,
  - Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps, dan
  - WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot

Selain itu, dalam uji coba ini juga digunakan perangkat lunak pembantu yaitu Microsoft Excel 2013.

5.2 Data Uji Coba

Data uji coba yang digunakan dalam uji coba ini merupakan data tanggapan yang dikeluarkan oleh departemen-departemen yang ada pada *database* Suara Warga Kota Kediri pada bulan tertentu, dimana data *training* dihimpun dari tanggapan-tanggapan

pada bulan Oktober 2014, sedangkan data *testing* dihimpun dari tanggapan-tanggapan pada tanggal 5 Januari – 20 Februari 2015. Data training yang digunakan akan dibagi menjadi 3 kategori, yaitu kategori baik, kategori netral, dan kategori tidak baik yang nantinyatanggapan yang belum diklasifikasi akan mendapatkan bobot penilaian/*scoring* berdasarkan hasil keluaran kategori setiap tanggapan tersebut.

Tanggapan-tanggapan ini dikelompokkan berdasarkan kategori yang dipisah menjadi data *training* dan data *testing*. Data *training* berjumlah 92 tanggapan, sedangkan data *testing* berjumlah 36 tanggapan. Daftar data *training* dan data *testing* yang digunakan sebagai data uji coba pada modul aplikasi ini dapat di lihat pada Tabel 5.1 dan Tabel 5.2.

Tabel 5.1 Daftar Data Training

No.	Kategori Tanggapan	Jumlah
1.	Baik	33
2.	Netral	26
3.	Tidak Baik	33
Total		**92**

Tabel 5.2 Daftar Data Testing

No.	Kategori Tanggapan	Jumlah
1.	Baik	8
2.	Netral	8
3.	Tidak Baik	20
Total		**36**

Jumlah data *training* lebih banyak dari jumlah data *testing* pada masing-masing kategori yang digunakan sebagai data uji pada Tugas Akhir ini. Hal ini dikarenakan data *training* akan digunakan sebagai data pembelajaran sistem dalam melakukan klasifikasi, sehingga sistem akan lebih mudah dalam mengidentifikasi suatu dokumen.

5.3 Skenario dan Hasil Uji Coba

Pada subbab ini dijelaskan skenario uji coba yang dilakukan dan hasil yang didapatkan. Uji coba terdiri dibagi menjadi dua yaitu uji coba klasifikasi dengan algoritma Naïve Bayes dan uji coba fungsionalitas.

5.3.1 Uji Coba Klasifikasi dengan Algoritma Naïve Bayes

Uji coba ini dilakukan untuk mengetahui keberhasilan proses klasifikasi tanggapan dengan menggunakan algoritma Naïve Bayes. Keberhasilan proses klasifikasi ini dapat dilihat pada hasil evaluasi dengan perhitungan akurasi. Perhitungan akurasi diukur dengan menghitung persentase kebenaran algoritma mengategorikan tanggapan dengan cara membandingkan kategori yang dihasilkan algoritma dengan data *ground truth* yang dihimpun dari hasil kuesioner yang telah diisi oleh responden. Untuk kebutuhan data *ground truth*, kuesioner yang diisi responden berisi tanggapan-tanggapan yang digunakan untuk *dataset* pada Tugas Akhir ini dengan opsi kategori Baik, Netral, atau Tidak Baik yang harus dipilih oleh responden. Responden yang mengisi kuesioner ini adalah petugas pada Pemerintahan Kota Kediri. Bukti pengisian kuesioner ini dapat dilihat pada LAMPIRAN C BERITA ACARA KUESIONER.

Tabel 5.3 merupakan hasil proses klasifikasi 36 tanggapan dengan menggunakan algoritma Naïve Bayes. Dari 36 data *testing* yang digunakan pada Tugas Akhir ini, ada sebanyak 7 tanggapan yang terklasifikasi salah, artinya tidak sesuai dengan kategori seharusnya menurut *ground truth*.

Tabel 5.3 Kategori Tanggapan Sebelum dan Sesudah Klasifikasi

No.	ID Data *Testing*	Kategori Seharusnya Menurut *Ground Truth*	Kategori yang Dihasilkan
1.	1	Netral	Tidak Baik
2.	2	Netral	Netral

3.	3	Baik	Tidak Baik
4.	4	Netral	Baik
5.	5	Netral	Netral
6.	6	Netral	Netral
7.	7	Baik	Baik
8.	8	Baik	Baik
…	…	...	…
34.	34	Tidak Baik	Tidak Baik
35.	35	Tidak Baik	Tidak Baik
36.	36	Tidak Baik	Tidak Baik

Dari 36 data testing yang digunakan pada Tugas Akhir ini dan dengan menggunakan perhitungan akurasi, dimana sebanyak 7 dari 36 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 80,56% dengan porsi data *testing* sebesar 28% dari total 128 *dataset* yang digunakan.

Selain menghitung akurasi pada 36 tanggapan data *testing*, uji coba juga akan dilakukan pada keseluruhan *dataset* yang berjumlah 128 buah tanggapan yang digunakan pada Tugas Akhir ini menggunakan metode *K-Fold Cross-Validation.* Dari 128 buah tanggapan, akan dibagi menjadi 8 *subset* sama rata yang masing-masing *subset*-nya berjumlah 16 buah tanggapan. Kemudian *dataset* akan diklasifikasi sebanyak 8 kali, dengan 1 *subset* bergantian menjadi data *testing* sementara sisanya menjadi data *training*.

Tabel 5.4 merupakan hasil pengujian klasifikasi pada *subset* 1 dengan hasil, yaitu sebanyak 3 dari 16 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 81,25%.

Tabel 5.4 Hasil Klasifikasi *Subset* 1

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
1	1	Baik	Tidak Baik
	2	Netral	Baik
	3	Baik	Baik
	4	Baik	Baik
	5	Baik	Baik
	6	Tidak Baik	Tidak Baik
	7	Baik	Baik
	8	Tidak Baik	Tidak Baik
	9	Baik	Baik
	10	Baik	Baik
	11	Baik	Baik
	12	Baik	Baik
	13	Baik	Baik
	14	Netral	Netral
	15	Baik	Baik
	16	Netral	Tidak Baik

Tabel 5.5 merupakan hasil pengujian klasifikasi pada *subset* 2 dengan hasil, yaitu sebanyak 3 dari 16 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 81,25%.

Tabel 5.5 Hasil Klasifikasi *Subset* 2

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
2	17	Baik	Tidak Baik
	18	Tidak Baik	Tidak Baik
	19	Baik	Baik

	20	Baik	Baik
	21	Baik	Baik
	22	Baik	Baik
	23	Tidak Baik	Netral
	24	Tidak Baik	Tidak Baik
	25	Tidak Baik	Tidak Baik
	26	Baik	Baik
	27	Netral	Netral
	28	Tidak Baik	Tidak Baik
	29	Baik	Tidak Baik
	30	Netral	Netral
	31	Baik	Baik
	32	Baik	Baik

Tabel 5.6 merupakan hasil pengujian klasifikasi pada *subset* 3 dengan hasil, yaitu sebanyak 6 dari 16 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 62,50%.

Tabel 5.6 Hasil Klasifikasi *Subset* 3

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
3	33	Netral	Baik
	34	Netral	Baik
	35	Baik	Baik
	36	Netral	Netral
	37	Netral	Netral
	38	Netral	Netral
	39	Baik	Baik
	40	Tidak Baik	Tidak Baik
	41	Baik	Baik
	42	Netral	Tidak Baik
	43	Baik	Baik

	44	Baik	Baik
	45	Baik	Baik
	46	Netral	Baik
	47	Netral	Tidak Baik
	48	Netral	Baik

Tabel 5.7 merupakan hasil pengujian klasifikasi pada *subset* 4 dengan hasil, yaitu sebanyak 4 dari 16 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 75%.

Tabel 5.7 Hasil Klasifikasi *Subset* 4

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
4	49	Netral	Netral
	50	Tidak Baik	Tidak Baik
	51	Netral	Netral
	52	Netral	Tidak Baik
	53	Netral	Tidak Baik
	54	Tidak Baik	Tidak Baik
	55	Baik	Baik
	56	Netral	Tidak Baik
	57	Tidak Baik	Tidak Baik
	58	Baik	Baik
	59	Baik	Baik
	60	Baik	Baik
	61	Baik	Baik
	62	Baik	Tidak Baik
	63	Baik	Baik
	64	Baik	Baik

Tabel 5.8 merupakan hasil pengujian klasifikasi pada *subset* 5 dengan hasil, yaitu sebanyak 4 dari 16 tanggapan pada data

testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 75%.

Tabel 5.8 Hasil Klasifikasi *Subset* 5

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
5	65	Baik	Baik
	66	Tidak Baik	Tidak Baik
	67	Tidak Baik	Tidak Baik
	68	Tidak Baik	Tidak Baik
	69	Tidak Baik	Tidak Baik
	70	Tidak Baik	Tidak Baik
	71	Tidak Baik	Baik
	72	Tidak Baik	Tidak Baik
	73	Tidak Baik	Baik
	74	Tidak Baik	Tidak Baik
	75	Tidak Baik	Baik
	76	Tidak Baik	Tidak Baik
	77	Tidak Baik	Baik
	78	Baik	Baik
	79	Baik	Baik
	80	Tidak Baik	Tidak Baik

Tabel 5.9 merupakan hasil pengujian klasifikasi pada *subset* 6 dengan hasil, yaitu sebanyak 4 dari 16 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 75%.

Tabel 5.9 Hasil Klasifikasi *Subset* 6

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
6	81	Tidak Baik	Tidak Baik
	82	Tidak Baik	Tidak Baik
	83	Baik	Tidak Baik
	84	Tidak Baik	Tidak Baik
	85	Tidak Baik	Tidak Baik
	86	Baik	Tidak Baik
	87	Tidak Baik	Tidak Baik
	88	Tidak Baik	Tidak Baik
	89	Tidak Baik	Tidak Baik
	90	Tidak Baik	Tidak Baik
	91	Tidak Baik	Tidak Baik
	92	Tidak Baik	Tidak Baik
	93	Netral	Tidak Baik
	94	Netral	Netral
	95	Baik	Baik
	96	Netral	Baik

Tabel 5.10 merupakan hasil pengujian klasifikasi pada *subset* 7 dengan hasil, yaitu sebanyak 2 dari 16 tanggapan pada data testing terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 87,50%.

Tabel 5.10 Hasil Klasifikasi *Subset* 7

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
7	97	Netral	Netral
	98	Netral	Netral
	99	Baik	Baik

	100	Baik	Baik
	101	Netral	Netral
	102	Netral	Netral
	103	Baik	Baik
	104	Netral	Baik
	105	Baik	Baik
	106	Baik	Baik
	107	Baik	Baik
	108	Tidak Baik	Tidak Baik
	109	Tidak Baik	Tidak Baik
	110	Tidak Baik	Netral
	111	Tidak Baik	Tidak Baik
	112	Baik	Baik

Tabel 5.11 merupakan hasil pengujian klasifikasi pada *subset* 8 dengan hasil, yaitu tidak ada tanggapan pada data testing yang terklasifikasi salah, sehingga didapatkan tingkat akurasi mencapai 100%.

Tabel 5.11 Hasil Klasifikasi *Subset* 8

Subset	**ID Data *Testing***	**Kategori Seharusnya Menurut *Ground Truth***	**Kategori yang Dihasilkan**
8	113	Tidak Baik	Tidak Baik
	114	Tidak Baik	Tidak Baik
	115	Tidak Baik	Tidak Baik
	116	Tidak Baik	Tidak Baik
	117	Tidak Baik	Tidak Baik
	118	Tidak Baik	Tidak Baik
	119	Tidak Baik	Tidak Baik
	120	Tidak Baik	Tidak Baik
	121	Tidak Baik	Tidak Baik
	122	Tidak Baik	Tidak Baik
	123	Tidak Baik	Tidak Baik

	124	Tidak Baik	Tidak Baik
	125	Tidak Baik	Tidak Baik
	126	Tidak Baik	Tidak Baik
	127	Tidak Baik	Tidak Baik
	128	Tidak Baik	Tidak Baik

Setelah mendapat nilai akurasi dari setiap *subset* yang ada, maka akan dilakukan penilaian *cross-validation* terhadap akurasi model secara keseluruhan dihitung dengan mengambil rata-rata dari semua hasil akurasi *subset* dengan menggunakan Persamaan (2.5) seperti yang dijelaskan pada Bab II. Sehingga didapatkan tingkat akurasi dengan uji coba *K-Fold Cross-Validation* mencapai 79,68% seperti yang ditunjukkan pada Tabel 5.12.

Tabel 5.12 Tabel Rangkuman Hasil Uji Coba *K-Fold Cross-Validation*

Subset	**Akurasi (%)**
1	81,25%
2	81,25%
3	62,50%
4	75%
5	75%
6	75%
7	87,50%
8	100%
Rata-rata $(\bar{x})$	79,68%

Selanjutnya akan dihitung standar deviasi dengan terlebih dahulu menghitung nilai varians. Perhitungan ini dapat dilihat pada Tabel 5.13.

Tabel 5.13 Tabel Perhitungan Varians dan Standar Deviasi

Subset	x	$x - \overline{x}$	$(x - \overline{x})^2$
1	0,81	0,02	0,0004
2	0,81	0,02	0,0004
3	0,62	-0,17	0,0289
4	0,75	-0,04	0,0016
5	0,75	-0,04	0,0016
6	0,75	-0,04	0,0016
7	0,87	0,08	0,0064
8	1,00	0,21	0,0441
		Jumlah	0,085
		Varians	0,012
		Standar Deviasi	0,109

Dengan didapatkannya nilai standar deviasi sebesar 0,79 ± 0,109, dapat disimpulkan bahwa nilai *mean* pada sampel yang digunakan dalam uji coba ini dapat digunakan sebagai representasi dari keseluruhan data.

5.3.2 Uji Coba Fungsionalitas

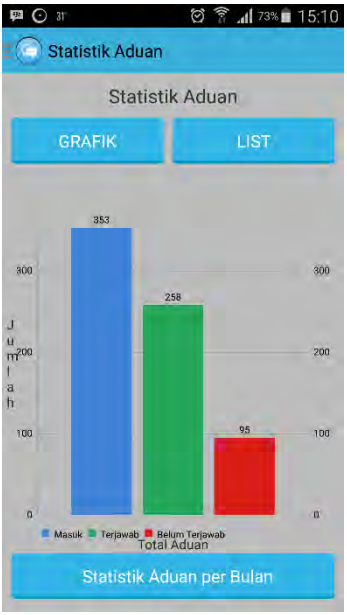
Uji coba fungsionalitas merupakan sebuah pengujian yang dilakukan terhadap jalannya fungsi-fungsi utama pada sistem yang telah dibuat. Pengujian dilakukan ke seluruh fungsi baik itu di sisi *client* maupun di sisi *server*. Uji coba fungsionalitas dilakukan terhadap semua *use case* yang telah dijelaskan pada subbab 3.2.1.

5.3.2.1 Uji Coba Menampilkan Statistik Aduan

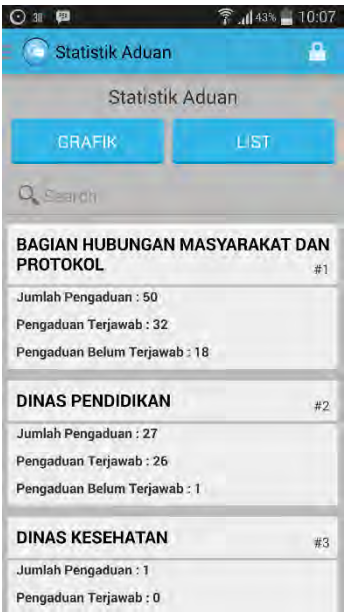
Uji coba ini dilakukan dengan mencoba menampilkan statistik aduan baik ke dalam bentuk diagram batang maupun daftar *listview,* sesuai dengan data yang telah tersimpan dalam *database*.

Tabel 5.14 Prosedur Uji Coba Menampilkan Statistik Aduan

ID	**UJ-01**
Referensi Use Case	**UC-01**
Nama	**Uji Coba Menampilkan Statistik Aduan**
Tujuan Uji Coba	Menguji fitur untuk menampilkan statistik aduan yang ada ke dalam bentuk diagram batang dan daftar *listview*
Kondisi Awal	Pengguna telah berada pada antarmuka statistik aduan
Data Masukan	-
Prosedur Pengujian	1. Memilih tombol 'List' untuk melihat statistik aduan dalam bentuk daftar *listview.* 2. Memilih tombol 'Grafik' untuk melihat statistik aduan dalam bentuk diagram batang.
Data Keluaran	Data statistik aduan dalam bentuk diagram batang dan daftar *listview*
Hasil Uji Coba	**Berhasil**



Gambar 5.2 Pengujian Menampilkan Statistik Aduan Dalam Bentuk Diagram Batang



Gambar 5.1 Pengujian Menampilkan Statistik Aduan Dalam Bentuk Daftar *Listview*

5.3.2.2 Uji Coba Melakukan Filter Aduan Perbulan

Uji coba ini dilakukan dengan mencoba menampilkan statistik aduan seluruh departemen pada bulan tertentu ke dalam bentuk diagram batang sesuai dengan data yang telah tersimpan dalam *database*. Pada saat pengguna berada pada antarmuka utama, yaitu antarmuka Statistik Aduan, penguna dapat menekan tombol 'Statistik Aduan per Bulan'. Setelah pengguna memilih bulan dan tahun yang diinginkan dan menekan tombol 'OK', maka modul aplikasi akan melakukan proses untuk menampilkan data statistik aduan pada bulan di tahun yang sudah dipilih tersebut.

Tabel 5.15 Prosedur Uji Coba Melakukan Filter Aduan Perbulan

ID	**UJ-02**
Referensi Use Case	**UC-02**
Nama	**Uji Coba Melakukan Filter Aduan Perbulan**
Tujuan Uji Coba	Menguji fitur untuk menampilkan statistik aduan seluruh departemen pada bulan tertentu ke dalam bentuk diagram batang
Kondisi Awal	Pengguna telah memilih bulan dan tahun yang diinginkan
Data Masukan	Bulan 'Januari' dan tahun '2015'
Prosedur Pengujian	1. Memilih tombol 'Statistik Aduan per Bulan'. 2. Memilih bulan. 3. Memilih tahun. 4. Memilih tombol 'OK'.
Data Keluaran	Data statistik aduan dalam bentuk diagram batang pada bulan Januari tahun 2015
Hasil Uji Coba	**Berhasil**

Gambar 5.3 Pengujian Melakukan Filter Aduan Perbulan

5.3.2.3 Uji Coba Menampilkan Statistik Rerata Waktu Penyelesaian Aduan

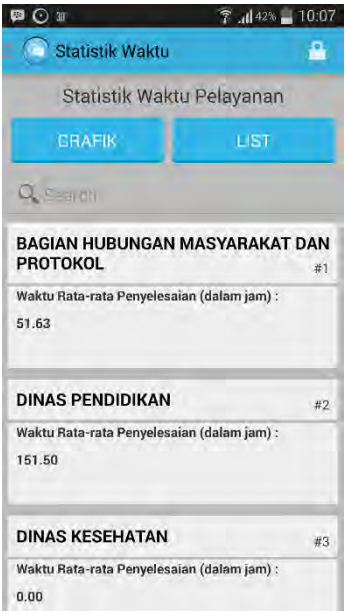
Uji coba ini dilakukan dengan mencoba menampilkan statistik rata-rata waktu penyelesaian aduan baik ke dalam bentuk diagram batang maupun daftar *listview,* sesuai dengan data yang telah tersimpan dalam *database*.

Tabel 5.16 Prosedur Uji Coba Menampilkan Statistik Rerata Waktu Penyelesaian Aduan

ID	**UJ-03**
Referensi Use Case	**UC-03**
Nama	**Uji Coba Menampilkan Statistik Rerata Waktu Penyelesaian Aduan**

Tujuan Uji Coba	Menguji fitur untuk menampilkan statistik rata-rata waktu penyelesaian aduan oleh departemen yang ada ke dalam bentuk diagram batang dan daftar *listview*
Kondisi Awal	Pengguna telah berada pada antarmuka statistik waktu
Data Masukan	-
Prosedur Pengujian	1. Memilih tombol 'List' untuk melihat statistik rata-rata waktu penyelesaian aduan departemen dalam bentuk daftar *listview.* 2. Memilih tombol 'Grafik' untuk melihat statistik rata-rata waktu penyelesaian aduan departemen dalam bentuk diagram batang.
Data Keluaran	Data statistik rata-rata waktu penyelesaian aduan departemen dalam bentuk diagram batang dan daftar *listview*
Hasil Uji Coba	**Berhasil**

Gambar 5.5 Pengujian Menampilkan Statistik Rerata Waktu Dalam Bentuk Diagram Batang



Gambar 5.4 Pengujian Menampilkan Statistik Rerata Waktu Dalam Bentuk Daftar *Listview*

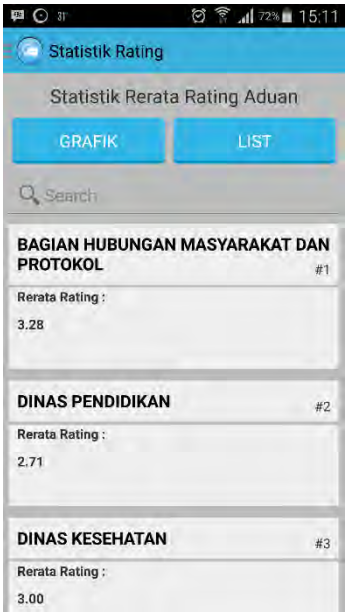
5.3.2.4 Uji Coba Menampilkan Statistik Rerata *Rating* Departemen

Uji coba ini dilakukan dengan mencoba menampilkan statistik rata-rata *rating* departemen baik ke dalam bentuk diagram batang maupun daftar *listview,* sesuai dengan data yang telah tersimpan dalam *database*.

Tabel 5.17 Prosedur Uji Coba Menampilkan Statistik Rerata *Rating* Departemen

ID	**UJ-04**
Referensi Use Case	**UC-04**
Nama	**Uji Coba Menampilkan Statistik Rerata *Rating* Departemen**
Tujuan Uji Coba	Menguji fitur untuk menampilkan statistik rata-rata *rating* departemen yang ada ke dalam bentuk diagram batang dan daftar *listview*
Kondisi Awal	Pengguna telah berada pada antarmuka statistik *rating*
Data Masukan	-
Prosedur Pengujian	1. Memilih tombol 'List' untuk melihat statistik rata-rata *rating* departemen dalam bentuk daftar *listview.* 2. Memilih tombol 'Grafik' untuk melihat statistik rata-rata *rating* departemen dalam bentuk diagram batang.
Data Keluaran	Data statistik rata-rata *rating* departemen dalam bentuk diagram batang dan daftar *listview*
Hasil Uji Coba	**Berhasil**

Gambar 5.7 Pengujian Menampilkan Statistik Rerata *Rating* Dalam Bentuk Diagram Batang



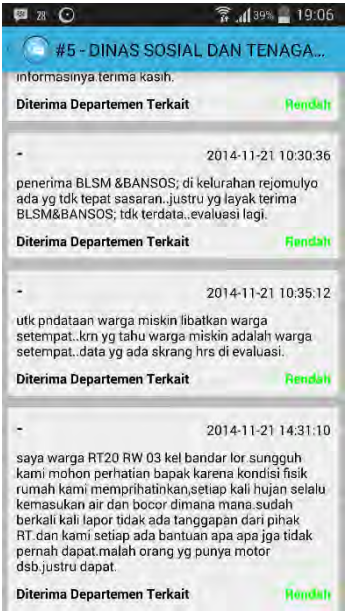
Gambar 5.6 Pengujian Menampilkan Statistik Rerata *Rating* Bentuk Daftar *Listview*

5.3.2.5 Uji Coba Menampilkan Aduan Belum Terjawab

Uji coba ini dilakukan dengan mencoba menampilkan daftar aduan-aduan yang belum dijawab oleh suatu departemen sesuai dengan data yang telah tersimpan dalam *database*. Pada saat pengguna berada pada antarmuka utama, yaitu antarmuka Statistik Aduan, pengguna dapat memilih departemen yang ingin dilihat aduan yang belum dijawabnya. Setelah pengguna memilih departemen yang diinginkan, maka modul aplikasi akan melakukan proses untuk menampilkan aduan-aduan belum terjawab dari departemen yang sudah dipilih tersebut.

Tabel 5.18 Prosedur Uji Coba Menampilkan Aduan Belum Terjawab

ID	**UJ-05**
Referensi Use Case	**UC-05**
Nama	**Uji Coba Menampilkan Aduan Belum Terjawab**
Tujuan Uji Coba	Menguji fitur untuk menampilkan aduan-aduan belum terjawab dari departemen tertentu
Kondisi Awal	Pengguna telah memilih departemen yang diinginkan
Data Masukan	Departemen 'Dinas Sosial dan Tenaga Kerja'
Prosedur Pengujian	1. Memilih tombol 'List' pada antarmuka Statistik Aduan. 2. Memilih departemen.
Data Keluaran	Data aduan-aduan belum terjawab milik Dinas Sosial dan Tenaga Kerja
Hasil Uji Coba	**Berhasil**



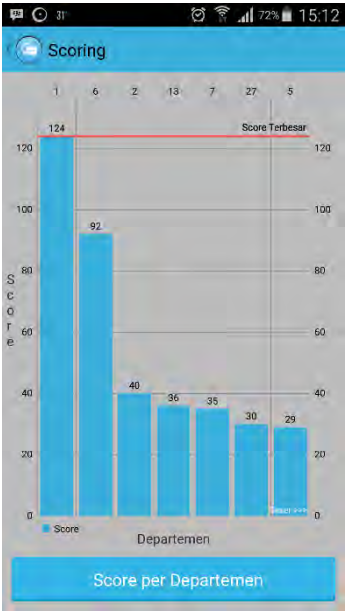
Gambar 5.8 Pengujian Menampilkan Aduan Belum Terjawab

5.3.2.6 Uji Coba Menampilkan Nilai Departemen

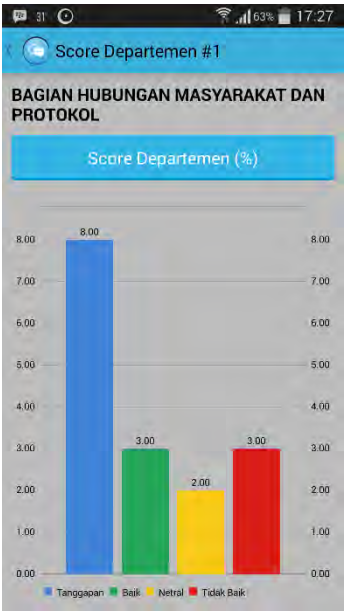
Uji coba ini dilakukan dengan mencoba menampilkan statistik nilai departemen sesuai dengan data yang telah tersimpan dalam *database,* baik seluruh departemen maupun untuk satu departemen yang dipilih. Nilai departemen diperoleh dari hasil klasifikasi terhadap tanggapan milik departemen tersebut. Pada saat pengguna berada pada antarmuka Tanggapan Departemen, pengguna dapat memilih opsi '*View Score*' dengan cara menekan tombol menu pada perangkat *smartphone.* Kemudian modul aplikasi akan menampilkan statistik nilai departemen dalam bentuk diagram batang.

Tabel 5.19 Prosedur Uji Coba Menampilkan Nilai Departemen

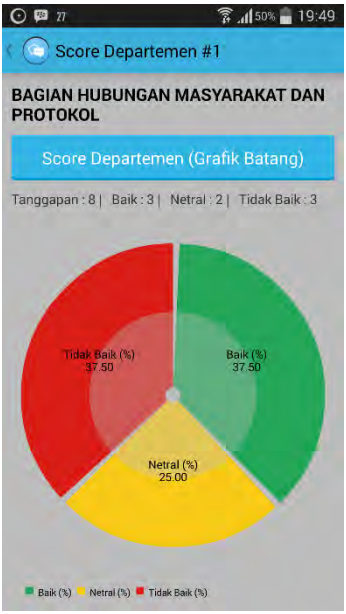
ID	**UJ-06**
Referensi Use Case	**UC-06**
Nama	**Uji Coba Menampilkan Nilai Departemen**
Tujuan Uji Coba	Menguji fitur untuk menampilkan statistik nilai departemen
Kondisi Awal	Pengguna telah berada pada antarmuka Tanggapan Departemen
Data Masukan	Departemen 'Bagian Hubungan Masyarakat dan Protokol' (Untuk melihat nilai perdepartemen)
Prosedur Pengujian	1. Memilih opsi '*View Score*' dengan menekan tombol menu pada perangkat *smartphone* untuk melihat statistik nilai departemen 2. Memilih tombol '*Score* per Departemen' 3. Memilih departemen 4. Memilih tombol '*Score* Departemen (%)' untuk melihat statistik nilai departemen dalam bentuk diagram lingkaran. 5. Memilih tombol '*Score* Departemen (Grafik Batang)' untuk melihat statistik nilai departemen dalam bentuk diagram batang.
Data Keluaran	1. Data statistik nilai seluruh departemen yang ada dalam bentuk diagram batang 2. Data statistik nilai departemen Bagian Hubungan Masyarakat dan Protokol dalam bentuk diagram batang 3. Data statistik nilai departemen Bagian Hubungan Masyarakat dan Protokol dalam bentuk diagram lingkaran
Hasil Uji Coba	**Berhasil**



Gambar 5.9 Pengujian Menampilkan Nilai Departemen



Gambar 5.11 Pengujian Menampilkan Statistik Nilai Departemen Bagian Hubungan Masyarakat dan Protokol Dalam Bentuk Diagram Batang



Gambar 5.10 Pengujian Menampilkan Statistik Nilai Departemen Bagian Hubungan Masyarakat dan Protokol Dalam Bentuk Diagram Lingkaran

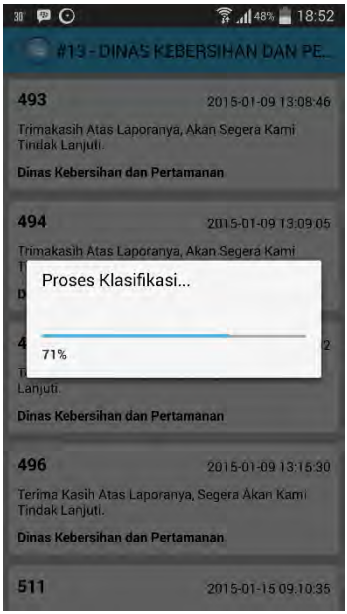
5.3.2.7 Uji Coba Melakukan Klasifikasi Tanggapan

Uji coba ini dilakukan dengan mencoba mengklasifikasikan tanggapan suatu departemen baik satu persatu, maupun seluruh tanggapan pada departemen tersebut secara langsung.

Tabel 5.20 Prosedur Uji Coba Melakukan Klasifikasi Tanggapan

ID	**UJ-07**
Referensi Use Case	**UC-07**
Nama	**Uji Coba Melakukan Klasifikasi Tanggapan**
Tujuan Uji Coba	Menguji fitur untuk melakukan proses klasifikasi tanggapan menggunakan algoritma Naïve Bayes
Kondisi Awal	Pengguna telah berada pada antarmuka Tanggapan Departemen
Data Masukan	Tanggapan dari departemen 'Bagian Hubungan Masyarakat dan Protokol'
Prosedur Pengujian	1. Memilih tanggapan yang ingin diklasifikasikan 2. Memilih opsi 'Klasifikasi Semua' dengan menekan tombol menu pada perangkat *smartphone*
Data Keluaran	Data tanggapan yang sudah memiliki kategori(Baik/Netral/Tidak Baik)
Hasil Uji Coba	**Berhasil**

Gambar 5.13 Pengujian Melakukan Klasifikasi Satu Tanggapan



Gambar 5.12 Pengujian Melakukan Klasifikasi Seluruh Tanggapan

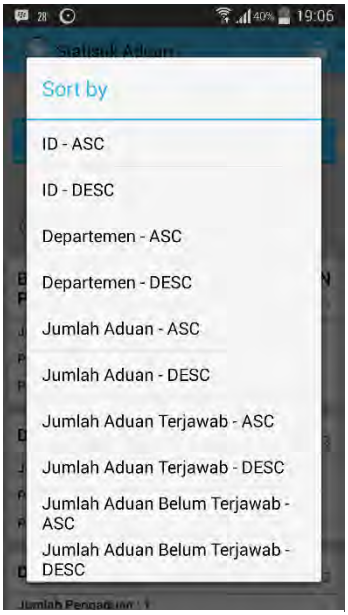
5.3.2.8 Uji Coba Mengurutkan Departemen

Uji coba ini dilakukan dengan mencoba mengurutkan daftar *listview* departemen dengan salah satu opsi pengurutan yang tersedia.

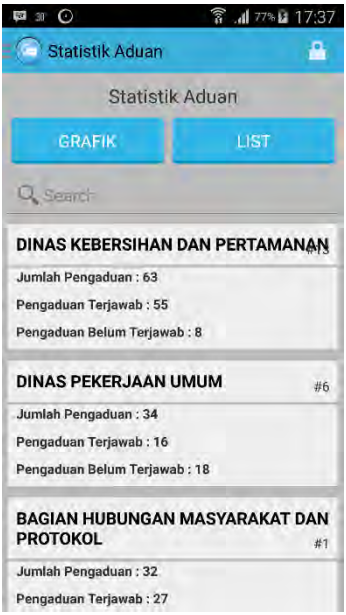
Tabel 5.21 Prosedur Uji Coba Mengurutkan Departemen

ID	**UJ-08**
Referensi Use Case	**UC-08**
Nama	**Uji Coba Mengurutkan Departemen**

Tujuan Uji Coba	Menguji fitur untuk melakukan proses pengurutan daftar departemen
Kondisi Awal	Pengguna telah memilih opsi '*Sort*' dengan cara menekan tombol menu pada *smartphone*
Data Masukan	opsi pengurutan 'Jumlah Aduan - DESC'
Prosedur Pengujian	1. Memilih opsi pengurutan dari *dialog window* yang muncul.
Data Keluaran	Data departemen pada daftar *listview* terurut berdasarkan departemen dengan jumlah aduan terbanyak sampai tersedikit
Hasil Uji Coba	**Berhasil**



Gambar 5.15 Pengujian Menampilkan Opsi-opsi Pengurutan



Gambar 5.14 Pengujian Mengurutkan Departemen

5.3.2.9 Uji Coba Mencari Departmen

Uji coba ini dilakukan dengan mencoba mencari departemen pada daftar *listview* departemen berdasarkan kata kunci yang dimasukkan pengguna. Kata kunci dapat berupa nomor departemen, nama departemen, jumlah aduan, rata-rata waktu, rata-rata rating, jumlah tanggapan, dan atribut-atribut departemen lainnya yang ada pada daftar *listview* yang ingin dicari.

Tabel 5.22 Prosedur Uji Coba Mencari Departemen

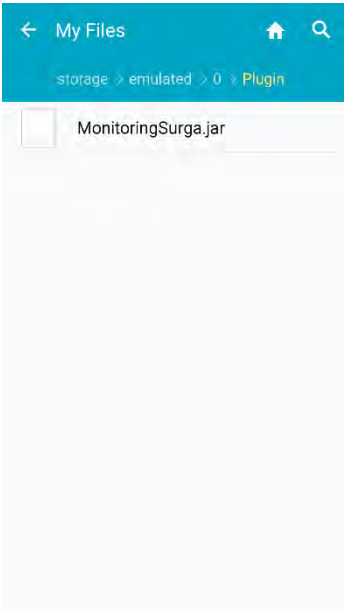
ID	**UJ-09**
Referensi Use Case	**UC-09**
Nama	**Uji Coba Mencari Departemen**
Tujuan Uji Coba	Menguji fitur untuk mencari departemen yang diinginkan berdasarkan kata kunci yang dimasukkan
Kondisi Awal	Pengguna telah berada pada antarmuka yang memiliki daftar *listview* departemen
Data Masukan	kata kunci pencarian ‘hubung’
Prosedur Pengujian	1. Memasukkan kata kunci pada kolom yang tersedia.
Data Keluaran	Data departemen pada daftar *listview* yang mengandung kata ‘hubung’ pada salah satu atributnya (dalam kasus ini, nama)
Hasil Uji Coba	**Berhasil**

Gambar 5.16 Pengujian Mencari Departemen

5.3.3 Uji Coba Modularitas

Uji coba modularitas merupakan sebuah pengujian yang dilakukan untuk mengetahui kerberhasilan modul aplikasi pada Tugas Akhir ini apakah dapat diimplementasikan pada aplikasi utama, yaitu aplikasi Suara Warga Kota Kediri. Uji coba ini dilakukan dengan beberapa langkah agar modul aplikasi dapat diakses serta dioperasikan oleh aplikasi utama. Langkah-langkah tersebut diantaranya sebagai berikut.

1. Eksport kelas PluginMonitoring menjadi *file* .jar.
2. *Compile file* .jar tersebut menggunakan dex untuk menghasilkan keluaran berupa *file* Classes.dex.
3. Masukkan Classes.dex tersebut ke dalam *file* .jar.
4. Masukkan *file* .jar ke folder Plugin di SD Card, seperti yang ditunjukkan pada Gambar 5.18.



Gambar 5.18 Folder Plugin Pada Perangkat Bergerak

Gambar 5.17 Hasil Pengujian Proses Modularitas

5.4 Evaluasi

Dari uji coba yang telah dilakukan, diketahui bahwa sistem telah dapat bekerja dengan baik dan benar dalam menjalankan fungsionalitasnya.

Pada uji coba klasifikasi yang dinilai berdasarkan perhitungan akurasi, sistem menghasilkan nilai akurasi mencapai 80,56% dengan porsi data *testing* sebesar 28% dari total 128 *dataset* yang digunakan. Dari 36 tanggapan yang digunakan sebagai data *testing*, ada sebanyak 7 tanggapan yang terklasifikasi tidak sesuai dengan kategori seharusnya menurut *ground truth*, dengan rincian 2 tanggapan salah klasifikasi pada kategori yang seharusnya berkategori baik, 3 tanggapan salah klasifikasi pada

kategori yang seharusnya berkategori netral, dan 2 tanggapan salah klasifikasi pada kategori yang seharusnya berkategori tidak baik. Sehingga dengan perhitungan akurasi data *testing* didapat nilai akurasi mencapai 80,56%. Sedangkan pada uji coba klasifikasi menggunakan metode *K-Fold Cross-Validation* pada 128 buah tanggapan pada *dataset* menghasilkan nilai akurasi mencapai 79,68% dengan nilai k=8 dan pada masing-masing *subset* memiliki jumlah 16 buah tanggapan dengan standar deviasi sebesar 0,79 ± 0,109.

Kemudian, berdasarkan hasil pengujian fungsionalitas, seluruh skenario berhasil dilakukan. Evaluasi terhadap pengujian fungsionalitas yang telah dilaksanakan dijelaskan sebagai berikut.

1. Fungsionalitas menampilkan statistik aduan berjalan sesuai dengan yang diharapkan.
2. Fungsionalitas melakukan filter aduan perbulan berjalan sesuai dengan yang diharapkan.
3. Fungsionalitas menampilkan statistik rerata waktu penyelesaian aduan berjalan sesuai dengan yang diharapkan.
4. Fungsionalitas menampilkan statistik rerata rating departemen berjalan sesuai dengan yang diharapkan.
5. Fungsionalitas menampilkan aduan belum terjawab berjalan sesuai dengan yang diharapkan.
6. Fungsionalitas menampilkan nilai departemen berjalan sesuai dengan yang diharapkan.
7. Fungsionalitas melakukan klasifikasi tanggapan berjalan sesuai dengan yang diharapkan.
8. Fungsionalitas mengurutkan departemen berjalan sesuai dengan yang diharapkan.
9. Fungsionalitas mencari departemen berjalan sesuai dengan yang diharapkan.

Sementara pada uji coba modularitas, modul aplikasi pada Tugas Akhir ini telah berhasil mengimplementasikan kelas *interface* dari aplikasi utama Suara Warga Kota Kediri, sehingga

modul aplikasi ini dapat diakses dan dioperasikan oleh aplikasi utama.

Dari keseluruhan hasil pengujian fungsionalitas pada modul aplikasi Tugas Akhir ini, seluruh skenario telah berhasil dilakukan. Modul aplikasi ini telah berjalan sesuai dengan yang diharapkan.

BAB VI
KESIMPULAN DAN SARAN

Dalam bab terakhir ini dijelaskan kesimpulan yang didapat dari pengerjaan Tugas Akhir ini beserta saran-saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut.

6.1 Kesimpulan

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil sebagai berikut.

1. Modul aplikasi yang dibangun pada Tugas Akhir ini dapat diimplementasikan pada aplikasi perangkat bergerak Suara Warga Kota Kediri.
2. Modul aplikasi yang dibangun pada Tugas Akhir ini dapat berjalan pada perangkat komunikasi bergerak dengan sistem operasi Android untuk melakukan *monitoring* aduan suara warga Kota Kediri.
3. Algoritma Naïve Bayes dapat diaplikasikan untuk melakukan kategorisasi atau pengelompokan tanggapan berbahasa Indonesia dengan tingkat akurasi ujicoba data *testing* mencapai 80,56% dan uji coba *K-Fold Cross-Validation* mencapai 79,68%.
4. Modul aplikasi yang dibangun pada Tugas Akhir ini dapat menampilkan data statistik aduan ke dalam bentuk diagram batang dengan menggunakan *library* MPAndroidChart dan ke dalam bentuk daftar *listview*.

6.2 Saran

Adapun saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut adalah sebagai berikut.

1. Perlu adanya riset untuk mengetahui apakah proses *stemming* Bahasa Indonesia memiliki pengaruh terhadap proses klasifikasi [4].

2. Untuk meningkatkan nilai akurasi pada klasifikasi tanggapan dalam Tugas Akhir ini, perlu adanya riset lebih lanjut untuk menentukan dasar kategorisasi tanggapan.
3. Solusi alternatif untuk lebih mempersingkat waktu yang dibutuhkan dalam proses klasifikasi tanggapan dengan cara melakukan proses perhitungan Naïve Bayes pada sisi *server*.
4. Meskipun pengguna Android sudah sangat marak, akan lebih baik lagi jika Tugas Akhir ini dapat diimplementasikan juga ke dalam *platform* lain seperti iOS dan Windows Phone.

DAFTAR PUSTAKA

[1] D. Mashudi, "Layanan Suara Warga di Kediri Kebanjiran Pengaduan," Surya Online, 11 October 2014. [Online]. Available: http://surabaya.tribunnews.com/2014/10/11/layanan-suara-warga-di-kediri-kebanjiran-pengaduan. [Accessed 2 July 2015].

[2] J. A. Sofyan, "Klasifikasi dalam Datamining," November 2012. [Online]. Available: http://www.publicmedialearning.com/discussion/view/241/klasifikasi-dalam-datamining. [Accessed 27 January 2015].

[3] P. Anugroho, I. Winarno and N. Rosyid, "Klasifikasi Email Spam Dengan Metode Naïve Bayes Classifier Menggunakan Java Programming," *JURNAL TEKNIK POMITS,* vol. I, 2010.

[4] F. Z. Tala, "A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia," *Institute for Logic, Language and Computation Universiteit van Amsterdam The Netherlands,* 2003.

[5] J. Samodra, S. Sumpeno and M. Hariadi, "Klasifikasi Dokumen Teks Berbahasa Indonesia dengan Menggunakan Naïve Bayes," *Seminar Nasional Electrical, Informatics, and It's Education,* 2009.

[6] A. Rozaq, A. Z. Arifin and D. Purwitasari, "Klasifikasi Dokumen Teks Berbahasa Arab Menggunakan Algoritma Naive Bayes," *JURNAL TEKNIK POMITS,* vol. I, 2011.

[7] D. Sano, "Metode-metode dalam Data Mining - Seri Data Mining for Business Intelligence (6)," August 2013. [Online]. Available: http://beritati.blogspot.com/2013/08/seri-data-mining-for-business_28.html. [Accessed 27 June 2015].

[8] Y. Yuliandra, "Standard Deviasi atau Standard Error?," 5 July 2012. [Online]. Available: https://yorijuly14.wordpress.com/2012/07/05/standard-deviasi-atau-standard-error/. [Accessed 1 July 2015].

[9] B. Jateng, "Black Box Testing dan Contoh Pengujian Black Box," [Online]. Available: http://dasar-pendidikan.blogspot.com/2013/06/black-box-testing-dan-contoh-pengujian.html. [Accessed 10 March 2015].

[10] A. Uva, "Tentang Android," 16 January 2012. [Online]. Available: https://adiebroz2.wordpress.com/2012/01/16/tentang-android/. [Accessed 25 June 2015].

[11] P. Jahoda, "MPAndroidChart," [Online]. Available: https://github.com/PhilJay/MPAndroidChart. [Accessed 25 June 2015].

[12] Hamdani, "Apa itu Web Service?," 15 July 2011. [Online]. Available: http://hamdani.blog.ugm.ac.id/2011/07/15/apa-itu-web-service/. [Accessed 18 December 2014].

BAB VI
KESIMPULAN DAN SARAN

Dalam bab terakhir ini dijelaskan kesimpulan yang didapat dari pengerjaan Tugas Akhir ini beserta saran-saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut.

6.1 Kesimpulan

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil sebagai berikut.

1. Modul aplikasi yang dibangun pada Tugas Akhir ini dapat diimplementasikan pada aplikasi perangkat bergerak Suara Warga Kota Kediri.
2. Modul aplikasi yang dibangun pada Tugas Akhir ini dapat berjalan pada perangkat komunikasi bergerak dengan sistem operasi Android untuk melakukan *monitoring* aduan suara warga Kota Kediri.
3. Algoritma Naïve Bayes dapat diaplikasikan untuk melakukan kategorisasi atau pengelompokan tanggapan berbahasa Indonesia dengan tingkat akurasi ujicoba data *testing* mencapai 80,56% dan uji coba *K-Fold Cross-Validation* mencapai 79,68%.
4. Modul aplikasi yang dibangun pada Tugas Akhir ini dapat menampilkan data statistik aduan ke dalam bentuk diagram batang dengan menggunakan *library* MPAndroidChart dan ke dalam bentuk daftar *listview*.

6.2 Saran

Adapun saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut adalah sebagai berikut.

1. Perlu adanya riset untuk mengetahui apakah proses *stemming* Bahasa Indonesia memiliki pengaruh terhadap proses klasifikasi [4].

2. Untuk meningkatkan nilai akurasi pada klasifikasi tanggapan dalam Tugas Akhir ini, perlu adanya riset lebih lanjut untuk menentukan dasar kategorisasi tanggapan.
3. Solusi alternatif untuk lebih mempersingkat waktu yang dibutuhkan dalam proses klasifikasi tanggapan dengan cara melakukan proses perhitungan Naïve Bayes pada sisi *server*.
4. Meskipun pengguna Android sudah sangat marak, akan lebih baik lagi jika Tugas Akhir ini dapat diimplementasikan juga ke dalam *platform* lain seperti iOS dan Windows Phone.

LAMPIRAN A
PERANCANGAN DATA

Pada subbab ini dijelaskan tentang rancangan basis data yang akan digunakan pada Tugas Akhir ini. Basis data yang digunakan merupakan basis data pada sistem Suara Warga Kota Kediri yang dibangun menggunakan RDBMS MySQL.

1. Tabel aduan

Tabel aduan digunakan untuk menyimpan data seluruh aduan yang masuk ke sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu buah aduan dari warga. Atribut pada tabel ini dijelaskan pada Tabel A.1.

Tabel A.1 Tabel Aduan

Atribut	**Tipe**	***Null***	***Default***	**Keterangan**
id_aduan	int	Tidak	*autoincrement*	PK
no_identitas	varchar (45)	Ya	*null*	ktp pengadu
nama	varchar (45)	Ya	*null*	nama pengadu
alamat	varchar (45)	Ya	*null*	alamat pengadu
email	varchar (45)	Ya	*null*	email pengadu
no_hp	varchar (45)	Ya	*null*	no. telepon pengadu
tgl_lahir	date	Ya	*null*	tanggal lahir pengadu
topik	varchar (45)	Ya	*null*	topik aduan
isi	text	Ya	*null*	isi aduan
waktu	time stamp	Ya	*null*	waktu aduan ditulis

Atribut	**Tipe**	***Null***	***Default***	**Keterangan**
alamat_ip	varchar (45)	Ya	*null*	alamat ip pengadu
user_agent	varchar (2048)	Ya	*null*	*browser* pengadu jika aduan dikirim via *website*
longitude	varchar (45)	Ya	*null*	bujur informasi peta
latitude	varchar (45)	Ya	*null*	lintang informasi peta
rating	int	Ya	0	rating aduan
via_sms	tinyint	Ya	*null*	dikirim melalui sms
via_petugas	int	Ya	*null*	dikirim melalui petugas
info	int	Tidak	0	informasi aduan
info_detail	text	Ya	*null*	informasi detail aduan
spam	tinyint	Tidak	0	penanda jika aduan termasuk spam

2. Tabel detail_aduan

Tabel detail_aduan digunakan untuk menyimpan data isi aduan dan tanggapan dari petugas yang ada pada sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu buah aduan atau tanggapan. Atribut pada tabel ini dijelaskan pada Tabel A.2.

Tabel A.2 Tabel detail_aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_detail_aduan	int	Tidak	*autoincrement*	PK
isi_detail	text	Ya	*null*	isi aduan atau tanggapan
waktu_detail	datetime	Ya	*null*	waktu aduan atau tanggapan ditulis

3. Tabel petugas

Tabel petugas digunakan untuk menyimpan data petugas yang memiliki akun pada sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu akun milik petugas. Atribut pada tabel ini dijelaskan pada Tabel A.3.

Tabel A.3 Tabel petugas

Atribut	Tipe	*Null*	*Default*	Keterangan
id_petugas	int	Tidak	*autoincrement*	PK
username_petugas	varchar (45)	Ya	*null*	username petugas
password_petugas	varchar (45)	Ya	*null*	password petugas

Atribut	Tipe	*Null*	*Default*	Keterangan
nama_ petugas	varchar (45)	Ya	*null*	nama petugas
email_ petugas	varchar (45)	Ya	*null*	email petugas
no_hp_ petugas	varchar (45)	Ya	*null*	no. telepon petugas
jobdesc_ petugas	varchar (45)	Ya	*null*	deskripsi petugas
bisa_sms	tinyint	Tidak	0	penanda jika petugas dapat mengirim sms

4. Tabel departemen

Tabel departemen digunakan untuk menyimpan data seluruh departemen yang ada pada struktur pemerintahan Kota Kediri. Dimana setiap baris pada tabel mewakili satu departemen. Atribut pada tabel ini dijelaskan pada Tabel A.4.

Tabel A.4 Tabel departemen

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ departemen	int	Tidak	*autoincrement*	PK
nama_ departemen	varchar (128)	Ya	*null*	nama departemen
nama_ kepala	varchar (128)	Ya	*null*	nama kepala departemen
no_hp	varchar (45)	Ya	*null*	no. telepon kepala departemen

apakah_ mitra	tinyint	Ya	*null*	penanda jika suatu departemen termasuk mitra

5. Tabel *role*

Tabel role digunakan untuk menyimpan data *role* atau peranan dari setiap petugas. *Role* ini akan berpengaruh pada hak akses yang dimiliki oleh suatu petugas dalam mengakses sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu jenis *role*. Atribut pada tabel ini dijelaskan pada Tabel A.5.

Tabel A.5 Tabel *role*

Atribut	Tipe	*Null*	*Default*	Keterangan
id_role	int	Tidak	-	PK
nama_role	varchar (45)	Ya	*null*	nama role
deskripsi_ role	text	Ya	*null*	deskripsi role

6. Tabel kategori

Tabel kategori digunakan untuk menyimpan data kategori-kategori aduan. Dimana setiap baris pada tabel mewakili satu kategori aduan. Atribut pada tabel ini dijelaskan pada Tabel A.6.

Tabel A.6 Tabel kategori

Atribut	Tipe	*Null*	*Default*	Keterangan
id_kategori	int	Tidak	*autoincrement*	PK
nama_ kategori	varchar (45)	Ya	*null*	nama kategori

7. Tabel pengadu

Tabel pengadu digunakan untuk menyimpan data seluruh warga kota Kediri. Tabel ini berfungsi untuk verifikasi apakah warga yang menulis aduan adalah warga Kediri atau bukan. Dimana setiap baris pada tabel mewakili satu pengadu. Atribut pada tabel ini dijelaskan pada Tabel A.7.

Tabel A.7 Tabel pengadu

Atribut	Tipe	*Null*	*Default*	Keterangan
nik	varchar (50)	Tidak	-	PK
nama_lgkp	varchar (50)	Ya	*null*	nama pengadu
jenis_klmn	tinyint (4)	Ya	*null*	jenis kelamin pengadu
tmpt_lhr	varchar (50)	Ya	*null*	tempat lahir pengadu
tgl_lhr	date	Ya	*null*	tanggal lahir pengadu
pddk_akh	int	Ya	*null*	pendidikan terakhir
jenis_pkrjn	int	Ya	*null*	pekerjaan pengadu
no_prop	int	Ya	*null*	propinsi pengadu
no_kab	int	Ya	*null*	kabupaten pengadu
no_kec	int	Ya	*null*	kecamatan pengadu
no_kel	int	Ya	*null*	kelurahan pengadu

8. Tabel status

Tabel status digunakan untuk menyimpan data status aduan yang ditulis warga (apakah aduan sudah diterima departemen atau sudah selesai ditanggapi). Dimana setiap baris pada tabel mewakili satu jenis status aduan. Atribut pada tabel ini dijelaskan pada Tabel A.8.

Tabel A.8 Tabel status

Atribut	Tipe	*Null*	*Default*	Keterangan
id_status	int	Tidak	*autoincrement*	PK
nama_status	varchar (45)	Ya	*null*	nama status
class_status	varchar (45)	Ya	*null*	kelas status

9. Tabel status_aduan

Tabel status_aduan digunakan untuk menyimpan data *log* jika terjadi perubahan status aduan di tabel aduan. Dimana setiap baris pada tabel mewakili satu *log* dari suatu aduan. Atribut pada tabel ini dijelaskan pada Tabel A.9.

Tabel A.9 Tabel aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_status_ aduan	int	Tidak	*autoincrement*	PK
waktu_ status_ aduan	varchar (45)	Ya	*null*	Waktu *log* aduan

10. Tabel prioritas

Tabel prioritas digunakan untuk menyimpan data tingkat prioritas aduan yang ditulis warga. Dimana setiap baris pada tabel mewakili satu tingkat prioritas aduan. Atribut pada tabel ini dijelaskan pada Tabel A.10.

Tabel A.10 Tabel prioritas

Atribut	Tipe	*Null*	*Default*	Keterangan
id_prioritas	int	Tidak	*autoincrement*	PK
nama_ prioritas	varchar (45)	Ya	*null*	nama tingkatan prioritas
class_ prioritas	varchar (45)	Ya	*null*	kelas prioritas

11. Tabel kelurahan

Tabel kelurahan digunakan untuk menyimpan data kelurahan yang ada di Kota Kediri. Dimana setiap baris pada tabel mewakili satu kelurahan. Atribut pada tabel ini dijelaskan pada Tabel A.11.

Tabel A.11 Tabel kelurahan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ kelurahan	int	Tidak	*autoincrement*	PK
nama_ kelurahan	varchar (45)	Ya	*null*	nama kelurahan

12. Tabel kecamatan

Tabel kecamatan digunakan untuk menyimpan data kecamatan yang ada di Kota Kediri. Dimana setiap baris pada tabel mewakili satu kecamatan. Atribut pada tabel ini dijelaskan pada Tabel A.12.

Tabel A.12 Tabel kecamatan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ kecamatan	int	Tidak	*autoincrement*	PK
nama_ kecamatan	varchar (45)	Ya	*null*	nama kecamatan

13. Tabel *upload*

Tabel *upload* digunakan untuk menyimpan data berupa *path* dari *file* yang disimpan di *server*. *File* yang dimaksud adalah lampiran yang diunggah warga ketika mengisi aduan melalui *website*. Dimana setiap baris pada tabel mewakili satu *path* dari *file* lampiran. Atribut pada tabel ini dijelaskan pada Tabel A.13.

Tabel A.13 Tabel *upload*

Atribut	Tipe	*Null*	*Default*	Keterangan
id_upload	varchar (128)	Tidak	-	PK
orig_name	varchar (256)	Ya	*null*	nama asli dari *file*
path_upload	varchar (256)	Ya	*null*	*path* dari *file*
file_type	varchar (256)	Ya	*null*	tipe *file*

14. Tabel sms_tidak_valid

Tabel sms_tidak_valid digunakan untuk menyimpan data pesan singkat yang masuk namun tidak valid. Kriteria tidak valid yang dimaksud adalah ketika pesan singkat tidak sesuai dengan format penulisan yang sudah ditentukan pemerintah Kota Kediri. Dimana setiap baris pada tabel mewakili satu pesan singkat yang tidak valid. Atribut pada tabel ini dijelaskan pada Tabel A.14.

Tabel A.14 Tabel sms_tidak_valid

Atribut	Tipe	*Null*	*Default*	Keterangan
id	int	Tidak	-	PK
nomor_ pengirim	varchar (20)	Ya	-	nomor pengirim pesan singkat
urutan	varchar (20)	Ya	-	urutan pesan singkat
waktu	timesta mp	Ya	*current timestamp*	waktu pesan singkat diterima
isi	text		-	isi pesan singkat

15. Tabel all_app

Tabel all_app digunakan untuk menyimpan *url* dari aplikasi *web* suara warga. Dimana setiap baris pada tabel mewakili satu buah *url*. Atribut pada tabel ini dijelaskan pada Tabel A.15.

Tabel A.15 Tabel all_app

Atribut	Tipe	*Null*	*Default*	Keterangan
id_all_app	int	Tidak	*autoincrement*	PK
url_app	varchar (256)	Ya	*null*	nama kategori
desc_app	varchar (45)	Ya	*null*	Deskripsi aplikasi

16. Tabel user_app

Tabel user_app digunakan untuk menyimpan *log* dari petugas yang mengakses aplikasi *web* suara warga. Dimana setiap baris pada tabel mewakili satu *user*. Atribut pada tabel ini dijelaskan pada Tabel A.16.

Tabel A.16 Tabel user_app

Atribut	Tipe	*Null*	*Default*	Keterangan
id_all_app	int	Tidak	*autoincrement*	PK
id_app	int	Tidak	-	*url web*
petugas	int	Tidak	-	Petugas yang mengakses *url*

17. Tabel tanggapan

Tabel tanggapan digunakan untuk menyimpan data hasil klasifikasi tanggapan yang dilakukan pada modul aplikasi Tugas Akhir ini. Hasil klasifikasi yang disimpan nantinya akan digunakan

untuk menghitung total *score*/nilai suatu departemen. *Score* yang diperoleh suatu departemen tergantung dari hasil kategori setiap tanggapan dari departemen tersebut. Dimana untuk tanggapan berkategori Baik akan mendapat poin 10, tanggapan berkategori Netral akan mendapat poin 5, dan tanggapan berkategori Tidak Baik akan mendapat poin 1. Tabel tanggapan ini disimpan pada *database* yang berbeda dari tabel-tabel lainnya yang berada pada *database* Suara Warga Kota Kediri. Setiap baris pada tabel mewakili satu tanggapan yang sudah diklasifikasi. Atribut pada tabel ini dijelaskan pada Tabel A.17.

Tabel A.17 Tabel tanggapn

Atribut	**Tipe**	***Null***	***Default***	**Keterangan**
id_ tanggapan	int	Tidak	-	PK
isi_ tanggapan	varchar (100)	Tidak	-	isi tanggapan
waktu_ tanggapan	varchar (100)	Tidak	-	waktu tanggapan diberikan
id_petugas	int	Tidak	-	id petugas yang memberi tanggapan
nama_ petugas	varchar (100)	Tidak	-	nama petugas yang memberi tanggapan
id_dept_ petugas	int	Tidak	-	id departemen dari petugas yang memberi tanggapan

kategori_ tanggapan	varchar (100)	Ya	*null*	kategori hasil klasifikasi tanggapan
poin_ tanggapan	int	Ya	*null*	poin tanggapan berdasarkan kategori tanggapan

[Halaman ini sengaja dikosongkan]

LAMPIRAN B
DAFTAR STOPWORDS

Pada subbab ini dilampirkan daftar *stopwords* yang digunakan pada Tugas Akhir ini. *Stopwords* diambil dari Appendix D pada paper "*A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*" oleh Fadillah Z. Tala [4].

Tabel B.1 Daftar *Stopwords*

No.	Kata
1	ada
2	adanya
3	adalah
4	adapun
5	agak
6	agaknya
7	agar
8	akan
9	akankah
10	akhirnya
11	aku
12	akulah
13	amat
14	amatlah
15	anda
16	andalah
17	antar
18	diantaranya
19	antara
20	antaranya

No.	Kata
21	diantara
22	apa
23	apaan
24	mengapa
25	apabila
26	apakah
27	apalagi
28	apatah
29	atau
30	ataukah
31	ataupun
32	bagai
33	bagaikan
34	sebagai
35	sebagainya
36	bagaimana
37	bagaimanapun
38	sebagaimana
39	bagaimanakah
40	bagi

No.	Kata
41	bahkan
42	bahwa
43	bahwasanya
44	sebaliknya
45	banyak
46	sebanyak
47	beberapa
48	seberapa
49	begini
50	beginian
51	beginikah
52	beginilah
53	sebegini
54	begitu
55	begitukah
56	begitulah
57	begitupun
58	sebegitu
59	belum
60	belumlah
61	sebelum
62	sebelumnya
63	sebenarnya
64	berapa
65	berapakah
66	berapalah
67	berapapun
68	betulkah

No.	Kata
69	sebetulnya
70	biasa
71	biasanya
72	bila
73	bilakah
74	bisa
75	bisakah
76	sebisanya
77	boleh
78	bolehkah
79	bolehlah
80	buat
81	bukan
82	bukankah
83	bukanlah
84	bukannya
85	cuma
86	percuma
87	dahulu
88	dalam
89	dan
90	dapat
91	dari
92	daripada
93	dekat
94	demi
95	demikian
96	demikianlah

No.	Kata
97	sedemikian
98	dengan
99	depan
100	di
101	dia
102	dialah
103	dini
104	diri
105	dirinya
106	terdiri
107	dong
108	dulu
109	enggak
110	enggaknya
111	entah
112	entahlah
113	terhadap
114	terhadapnya
115	hal
116	hampir
117	hanya
118	hanyalah
119	harus
120	haruslah
121	harusnya
122	seharusnya
123	hendak
124	hendaklah

No.	Kata
125	hendaknya
126	hingga
127	sehingga
128	ia
129	ialah
130	ibarat
131	ingin
132	inginkah
133	inginkan
134	ini
135	inikah
136	inilah
137	itu
138	itukah
139	itulah
140	jangan
141	jangankan
142	janganlah
143	jika
144	jikalau
145	juga
146	justru
147	kala
148	kalau
149	kalaulah
150	kalaupun
151	kalian
152	kami

No.	Kata
153	kamilah
154	kamu
155	kamulah
156	kan
157	kapan
158	kapankah
159	kapanpun
160	dikarenakan
161	karena
162	karenanya
163	ke
164	kecil
165	kemudian
166	kenapa
167	kepada
168	kepadanya
169	ketika
170	seketika
171	khususnya
172	kini
173	kinilah
174	kiranya
175	sekiranya
176	kita
177	kitalah
178	kok
179	lagi
180	lagian

No.	Kata
181	selagi
182	lah
183	lain
184	lainnya
185	melainkan
186	selaku
187	lalu
188	melalui
189	terlalu
190	lama
191	lamanya
192	selama
193	selama
194	selamanya
195	lebih
196	terlebih
197	bermacam
198	macam
199	semacam
200	maka
201	makanya
202	makin
203	malah
204	malahan
205	mampu
206	mampukah
207	mana
208	manakala

No.	Kata
209	manalagi
210	masih
211	masihkah
212	semasih
213	masing
214	mau
215	maupun
216	semaunya
217	memang
218	mereka
219	merekalah
220	meski
221	meskipun
222	semula
223	mungkin
224	mungkinkah
225	nah
226	namun
227	nanti
228	nantinya
229	nyaris
230	oleh
231	olehnya
232	seorang
233	seseorang
234	pada
235	padanya
236	padahal

No.	Kata
237	paling
238	sepanjang
239	pantas
240	sepantasnya
241	sepantasnyalah
242	para
243	pasti
244	pastilah
245	per
246	pernah
247	pula
248	pun
249	merupakan
250	rupanya
251	serupa
252	saat
253	saatnya
254	sesaat
255	saja
256	sajalah
257	saling
258	bersama
259	sama
260	sesama
261	sambil
262	sampai
263	sana
264	sangat

No.	Kata
265	sangatlah
266	saya
267	sayalah
268	se
269	sebab
270	sebabnya
271	sebuah
272	tersebut
273	tersebutlah
274	sedang
275	sedangkan
276	sedikit
277	sedikitnya
278	segala
279	segalanya
280	segera
281	sesegera
282	sejak
283	sejenak
284	sekali
285	sekalian
286	sekalipun
287	sesekali
288	sekaligus
289	sekarang
290	sekarang
291	sekitar
292	sekitarnya

No.	Kata
293	sela
294	selain
295	selalu
296	seluruh
297	seluruhnya
298	semakin
299	sementara
300	sempat
301	semua
302	semuanya
303	sendiri
304	sendirinya
305	seolah
306	seperti
307	sepertinya
308	sering
309	seringnya
310	serta
311	siapa
312	siapakah
313	siapapun
314	disini
315	disinilah
316	sini
317	sinilah
318	sesuatu
319	sesuatunya
320	suatu

No.	Kata
321	sesudah
322	sesudahnya
323	sudah
324	sudahkah
325	sudahlah
326	supaya
327	tadi
328	tadinya
329	tak
330	tanpa
331	setelah
332	telah
333	tentang
334	tentu
335	tentulah
336	tentunya
337	tertentu
338	seterusnya
339	tapi
340	tetapi
341	setiap
342	tiap
343	setidaknya
344	tidak
345	tidakkah

No.	Kata
346	tidaklah
347	toh
348	waduh
349	wah
350	wahai
351	sewaktu
352	walau
353	walaupun
354	wong
355	yaitu
356	yakni
357	yang

[Halaman ini sengaja dikosongkan]

LAMPIRAN C
BERITA ACARA KUESIONER

Pada subbab ini dilampirkan lampiran berita acara kuesioner sebagai bukti pengisian kuesioner http://bit.ly/surgaKediri untuk pengambilan *ground truth* yang diisi oleh responden petugas pemerintahan Kota Kediri.

DAFTAR RESPONDEN KUESIONER PANDANGAN AHLI TERHADAP KATEGORI TANGGAPAN ADUAN SUARA WARGA KOTA KEDIRI

Responden yang bertanda tangan di bawah ini telah mengisi kuesioner dengan judul "Kuesioner Pandangan Ahli Terhadap Kategori Tanggapan Aduan Suara Warga Kota Kediri" yang diakses pada halaman *web* http://bit.ly/surgaKediri, untuk memenuhi kebutuhan Tugas Akhir Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya dengan judul "Implementasi Modul Monitoring dan Klasifikasi Tanggapan Aduan dengan Metode Naive Bayes Pada Aplikasi Perangkat Bergerak Suara Warga Kota Kediri"

No.	Nama Responden	Jabatan	Tanda Tangan
1.	Nanang S.	PDE.	
2.	Herwin Z.	Humas	
3.	Cheppy	bindik.	

Gambar C.1 Berita Acara Kuesioner

[Halaman ini sengaja dikosongkan]

LAMPIRAN D
UJI COBA KLASIFIKASI

Pada subbab ini dilampirkan tabel pengujian klasifikasi algoritma Naïve Bayes dengan perhitungan akurasi. Perhitungan akurasi diukur dengan menghitung persentase kebenaran algoritma mengategorikan tanggapan dengan cara membandingkan kategori yang dihasilkan algoritma dengan data *ground truth* yang dihimpun dari hasil kuesioner yang telah diisi oleh responden.

Tabel D.1 Tabel Uji Coba Klasifikasi

No.	ID Data *Testing*	Kategori Seharusnya Menurut *Ground Truth*	Kategori yang Dihasilkan
1.	1	Netral	Tidak Baik
2.	2	Netral	Netral
3.	3	Baik	Tidak Baik
4.	4	Netral	Baik
5.	5	Netral	Netral
6.	6	Netral	Netral
7.	7	Baik	Baik
8.	8	Baik	Baik
9.	9	Netral	Netral
10.	10	Netral	Netral
11.	11	Baik	Baik
12.	12	Netral	Baik
13.	13	Baik	Baik
14.	14	Baik	Baik
15.	15	Baik	Baik
16.	16	Tidak Baik	Tidak Baik
17.	17	Tidak Baik	Tidak Baik
18.	18	Tidak Baik	Tidak Baik
19.	19	Tidak Baik	Netral
20.	20	Baik	Tidak Baik

21.	21	Tidak Baik	Baik
22.	22	Tidak Baik	Tidak Baik
23.	23	Tidak Baik	Tidak Baik
24.	24	Tidak Baik	Tidak Baik
25.	25	Tidak Baik	Tidak Baik
26.	26	Tidak Baik	Tidak Baik
27.	27	Tidak Baik	Tidak Baik
28.	28	Tidak Baik	Tidak Baik
29.	29	Tidak Baik	Tidak Baik
30.	30	Tidak Baik	Tidak Baik
31.	31	Tidak Baik	Tidak Baik
32.	32	Tidak Baik	Tidak Baik
33.	33	Tidak Baik	Tidak Baik
34.	34	Tidak Baik	Tidak Baik
35.	35	Tidak Baik	Tidak Baik
36.	36	Tidak Baik	Tidak Baik

BIODATA PENULIS

Mahardhika Rizki Bagaskoro, dilahirkan di Kota Bogor, Jawa Barat pada tanggal 20 Agustus 1993. Penulis adalah anak pertama dari dua bersaudara. Penulis telah menempuh pendidikan formal yaitu di SD Bina Insani Bogor (1999-2005), SMP Negeri 4 Bogor (2005-2008) dan SMA Negeri 5 Bogor (2008-2011). Setelah lulus dari SMA Negeri 5 Bogor pada tahun 2011, Penulis mengikuti ujian masuk ITS dan diterima di Jurusan Teknik Informatika ITS pada tahun 2011 dan terdaftar dengan NRP 5111100188. Di Jurusan Teknik Informatika ITS ini, Penulis mengambil Bidang Algoritma dan Pemrograman (AP). Penulis memiliki ketertarikan pada pengembangan *software* perangkat bergerak khususnya *platform* Android serta pengembangan *web*. Penulis dapat dihubungi melalui alamat *e-mail* di mrbagaskoro@gmail.com.